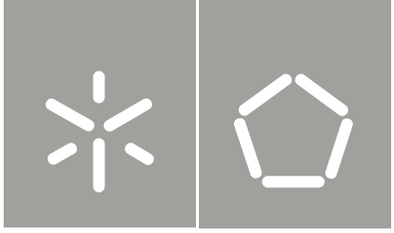
Universidade do Minho
Escola de Engenharia

José Pedro Ferreira Araújo

From Unimanual to Bimanual Manipulation in the Anthropomorphic Robot ARoS

José Pedro Ferreira Araújo
From Unimanual to Bimanual Manipulation in the Anthropomorphic Robot ARoS
UMinho | 2013

outubro de 2013



Universidade do Minho
Escola de Engenharia

José Pedro Ferreira Araújo

From Unimanual to Bimanual Manipulation in the Anthropomorphic Robot ARoS

Tese de Mestrado
Ciclo de Estudos Integrados Conducentes ao Grau de
Mestre em Engenharia Electrónica Industrial e Computadores

Trabalho efetuado sob a orientação de
Professora Doutora Estela Bicho
Professora Doutora Eliana Costa e Silva

outubro de 2013

DECLARAÇÃO

Nome: José Pedro Ferreira Araújo

Correio electrónico: jpara9@gmail.com

Tlm.: 253542111 / 919677778

Número do Bilhete de Identidade: 13546956

Título da dissertação:

From Unimanual to Bimanual Manipulation in the Anthropomorphic Robot ARoS

Ano de conclusão: 2013

Orientadores:

Estela Bicho

Eliana Costa e Silva

Designação do Mestrado:

Ciclo de Estudos Integrados Conducentes ao Grau de

Mestre em Engenharia Electrónica Industrial e Computadores

Área de Especialização: Automação, Controlo e Robótica

Escola: Escola de Engenharia

Departamento: Departamento de Electrónica Industrial

Guimarães, ___/___/_____

Assinatura: _____

Acknowledgments

This master's project would not be possible without the direct or indirect support of several people.

Firstly I would like to thank my scientific adviser, Professor Doctor Estela Bicho for having provided me this opportunity and for believing in me. Your constant support and availability contributed greatly to the development of my work. Your's technical and scientific knowledge were one of the bases for the success of this project. Many many thanks!

I also owe my deepest gratitude to my co-adviser, Doctor Eliana Oliveira da Costa e Silva. Thank you for being always available to answer my questions and for your patience to help me solve various problems during the different phases of this dissertation. Your support was trully crucial.

I want to thank Professor Doctor Fernanda Costa, from the Department of Mathematics and Applications at University of Minho for her immediate availability to help me understand the mathematical concepts of nonlinear optimization and for having me introduced to one of the programs that are in the basis of the results of this dissertation.

I am also indebted to my colleagues in the Laboratory of Mobile and Anthropomorphic Robots at University of Minho, Luis, Rui, Tiago, Emanuel, Carlos, Toni and Flora. Yours everyday support was very important, thanks to you all, one doubt never lasted too long. It is great to be part of this team. Thank you Luis for all the support and for implementing the interface of communication between **ARoS** and Matlab®. Special thanks to all the people who participated on the user-studies.

Many thanks to my colleague and friend Dário Machado who developed with me his dissertation, our conversations and work contributed in a very positive way for this work over the last year.

I would also like to show all my gratitude to my family for supporting me

and making me a better person everyday. To my parents, Ana Ferreira and José Araújo my brothers Filipe Araujo and José Araújo, my maternal grandparents, José Ferreira and Ana Freitas as well as my paternal grandparents Francisco Araujo and Emilia Oliveira. Special thanks to my mom for always being available to me despite the moment. Thank you for all the good advices, I am sure that this work would not be possible without you. By all the availability and interest I also want to thank António Xavier and Maria Amelia, thank you for all the kindness. Finally I want to make a very special acknowledgment to my girlfriend Catarina Xavier for all the support you have given me, in a personal and professional way. Thanks for reading this dissertation and for giving me great advices. Thank you for all the affection and for believing in me.

Abstract

The field of anthropomorphic robotics is unanimously recognized as one of the major challenges of current robotics. The reason is due to the fact that anthropomorphic robots are expected to co-exist and work cooperatively with human users in everyday environments where they may be needed. It is known that social human-robot interaction, as well as physical communication is easier if the robot has human form and behave similarly to human beings (Fong et al., 2002; Duffy, 2003; Schaal, 2007). This implies that the robot should possess cognition and human movements.

Considering that the majority of daily tasks and objects require bimanual manipulation (e.g. carry a tray, open a suitcase, cut bread), it is of outmost importance that humanoid robots are capable of cooperatively use both arms. Such tasks could either be performed by the robot itself or in collaboration with a human user.

Taking this significant need into consideration, this master thesis entitled "From unimanual to bimanual manipulation in the anthropomorphic robot ARoS" emerges, also integrating the European project PF7 Marie Curie "NETT - Neural Engineering Transformative Technologies". The main goal of this master project is to design a real-time optimized movement planning for unimanual motions. Only afterwards it will be possible the planning and control system design for bimanual movements for an anthropomorphic robotic system with *human-like* movement. This research project is under development at the Mobile and Anthropomorphic Robotics Laboratory (MARLab) of the Departament of Industrial Eletronics and the Algoritmi Center at the University of Minho (Campus Azurém).

With the intent of ease the interaction between the human user and the robot, and to increase the number of tasks that the robot can perform alone, it is necessary to add a second arm, which will reduce limitations and open a new

range of possibilities and bimanual applications. To coordinate two robotic arms with the main goal of executing a task in an independent or cooperative manner has been considered an enormous challenge in the development of personal robots. Some reasons for this perspective are mainly the elevated number of *degrees of freedom* (DOF) in the robotic arms that must be controlled, the unpredictability of daily tasks and the need to have a real-time system.

The advantages of a robot capable of executing bimanual tasks are very alluring, the main are highlighted bellow:

- The increase of range in applications. The second arm can hold an object while the other arm is in action, this will increase the capabilities of the robot in helping the human;
- The increase of the robot's workspace (e.g. An object might be out of range of one arm and within range to the second one)
- The increase of speed to which the task is executed.

Since the main goal is to be able to produce a practical and highly adaptable solution, the effort is being applied in order to have no pre-programmed movements in both robotic arms. Service robots in very dynamic environments should not be pre-programmed, therefore it will be possible to have a solution that is able to act after reading its surroundings and be directly adapted. It is extremely important the usage of such specific platforms to enlarge the knowledge in this exciting area and conclude with the improvement of humanity life quality. Simulations and tests were conducted in two different scenarios named has *"Toy-Vehicle"* and *"Space-Station"*. The results suggest that the above referred objectives and premises were accomplished, such has: **i)** Human-like movement, **ii)** Real-time planning and execution, **iii)** easy adaptation to dynamical scenarios cluttered with obstacles.

Resumo

A área da robótica antropomórfica é unanimemente reconhecida como um dos grandes desafios da robótica atual. Tal deve-se ao facto de que se pretende que os robôs antropomórficos coexistam e trabalhem em cooperação com os utilizadores humanos em todos os ambientes em que estes possam ser necessários. Sabe-se que a interação social humano-robô, bem como a comunicação física é facilitada se o robô possuir forma humana e se comportar de forma semelhante aos humanos (Fong et al., 2002; Duffy, 2003; Schaal, 2007). Isto implica que o robô possua cognição e movimentos humanos *(human like)*.

Considerando que a maioria das tarefas do dia-a-dia exigem manipulação bimanual (e.g. carregar uma bandeja, abrir uma mala, cortar pão), é importante que os robôs humanóides sejam capazes de cooperar com o humano utilizando ambos os braços. Tais tarefas podem ser realizadas pelo robô de forma independente ou em colaboração com o parceiro humano.

Tendo esta necessidade em consideração, foi proposto este projeto de mestrado intitulada "*De Manipulação Unimanual para Bimanual no Robô Antropomórfico ARoS*", integrado no projecto europeu PF7 Marie Curie " NETT - Neural Engineering Transformative Technologies".

O sistema robótico antropomórfico desenvolvido irá interagir com um parceiro humano, sendo capaz de, não só executar tarefas que o utilizador humano não é capaz de realizar devido a algum tipo de incapacidade, mas também, quando é vantajoso cooperar com o parceiro humano em diferentes tipos de situações (e.g. manipular uma garrafa, passá-la de uma mão para a outra e entregá-la ao humano).

Na anterior versão do **ARoS** vários problemas e limitações relacionados com à interação entre o robô e o utilizador humano, surgiram devido ao fato de que esta versão da plataforma robótica exibia apenas um manipulador.

Para facilitar a interação entre o utilizador humano e o **ARoS**, bem como

para aumentar o número de tarefas que o robô pode executar autonomamente, tornou-se claro que a adição de um segundo braço robótico iria reduzir limitações do sistema robótico e tornar possível uma série de novas aplicações bimanuais. Coordenar dois braços robóticos com o objetivo de executar tarefas de forma independente ou cooperativa é considerado um grande desafio no desenvolvimento de robôs pessoais. Algumas das razões que suportam esta perspectiva estão relacionadas com o elevado número de *graus de liberdade (DOF)* que braços robóticos antropomórficos possuem. Note-se também que a imprevisibilidade das tarefas do dia-a-dia dão ênfase à necessidade de se obter um sistema capaz de planear o movimento de ambos os braços em tempo real.

As vantagens de um robô capaz de executar tarefas bimanuais são por demais aliciantes entre as quais, pode ser destacadas as seguintes:

- O aumento da gama de aplicações. Um braço pode segurar num objeto, enquanto que o outro está a executar a ação relativamente ao primeiro, permitindo assim o aumento das possíveis aplicações do robô enquanto ajudante.
- O aumento do espaço de trabalho do robô (e.g. um objeto pode estar fora do alcance de um braço e dentro do alcance do outro braço)
- O aumento da velocidade com que a tarefa é executada.

Uma vez que se pretende obter uma solução adaptável, o projeto foi conduzido no sentido de não possuir movimentos pré-programado em ambos os braços robóticos. Esta premissa trará vantagens quando o **ARoS** for testado em ambientes dinâmicos onde os objetos não têem uma posição fixa, obtendo-se assim um sistema robótico capaz de se adaptar a mudanças na sua área de trabalho. Foram feitas simulações e testes em dois cenários distintos "Construção de um brinquedo" *(Toy Vehicle)* e "Estação Espacial" *(Space-Station)* com resultados que comprovam as premissas acima descritas, tais como: **i)** movimento idêntico ao do humano, **ii)** planeamento em tempo real, **iii)** adaptação a mudanças no meio, em ambientes com um elevado número de obstaculos.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 General Motivation and Objectives

1.1.1 Anthropomorphic robots, what to expect? How important are they now and in the future?

It is a human characteristic to look for ways to complete a task as easy and effectively as possible. Since the beginning, we affirm and conduct our technological evolution with this attitude in mind. We began by making choppers, spears and use fire to assist us hunting and later on we started seeking for means to have a sustainable agriculture and industry with the highest possible yield. The following step was to create mechanisms to help us (directly) in quotidian tasks, helping us have a safer and more comfortable life. Thus the first robots appeared starting by replacing us in uncomfortable, repetitive and potentially dangerous tasks (e.g. work on assembly lines for automobiles, mining, agriculture). Over the years technological development allowed the expansion of the spectrum of applications for robotic systems, and as a result today the use of robots has become natural. Their request from society was such that in recent years, their use has become natural and a basic trend to our well being and comfort (Coradeschi et al., 2006). As technology evolved, more advanced and complex robotic solutions arose, such

as:

- Robots for handling radioactive waste (Kim et al., 2006);
- Remote exploitation of seabed (Saltaren et al., 2007);
- Handles with high precision (Mohd Zubir and Shirinzadeh, 2009);
- Robots for space exploration as Robonaut2 (Diftler et al., 2011);
- Robot surgeons in Human-Robot Cooperation (Padoy and Hager, 2011);
- Robots to directly support the human in everyday tasks like the humanoid REEM-B (Tellez et al., 2008).

Industrial robots were the first solutions that visionary roboticists addressed (Tagliasco, 1991). Accordingly, in the 1860's and a lot due to the industrial revolution, the first industrial robots were placed in factories to replace humans in dangerous operating stations. This research area experienced a very rapid evolution due to human and monetary investment (Garcia et al., 2007), for this reason these robots were quickly gaining more flexibility and ability to perform harder tasks and therefore they started to be able to work in a wider range of areas within the industry. Over the past 45 years, the increase of areas applied to robotics has allowed the beginning of new research topics. One of those areas which has gained more relevance through the last few decades has been directed to address the personal needs of humans.

Currently, during the advent of the 21^{st} century, new needs have appeared and consequently, new markets outside the traditional ones (e.g. cleaning, construction and agriculture) have emerged such as personal service robots that provide personal and specialized attention. One of the sub-areas of robotics with the highest incidence of research is the robotic *manipulators* field and more specifically on how to plan robot movements to be directed to the human users and their purposes. A robotic manipulator may be defined as is a series of rigid members

connected to each other and designed to perform a task with his *End-Effector* (last rigid member).

As it is possible to observe in fig. 1.1, technological developments on manipulators have allowed their application to extrapolate to other contexts, more specific helping humans in tasks with increasingly higher complexity. Thus, the research field of humanoid robots emerged.

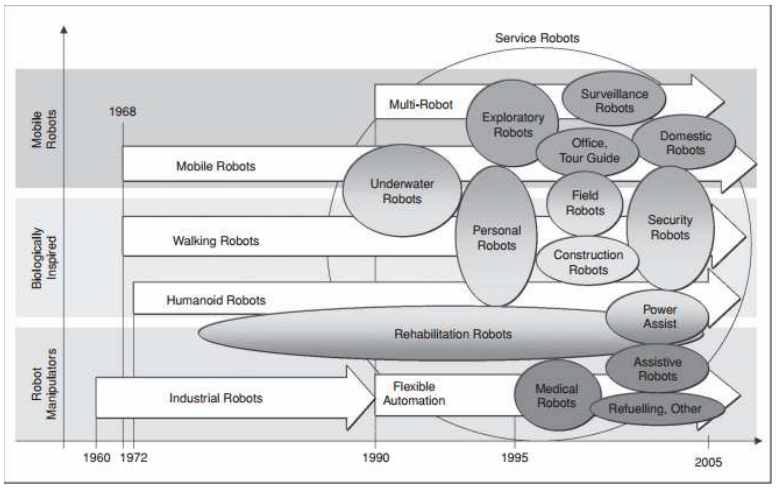


Figure 1.1: Evolution over time of research in robotics applied to service areas (Garcia et al., 2007).

The research field of humanoid robots has been recognized as one of the robotics area where research is more complicated and slower, mostly due to the responsibility and difficulty that the associated tasks entail. However, researchers around the world still remain motivated to work and to find more appropriate solutions.

The robot must have a range of abilities and skills similar to those that humans can apply. Following this, it will become possible for the robot to perform tasks such as rehabilitation, first aid, help elder users to control their habits, housework, companion to people with special needs, among others. Their role will be increasingly closer to that which is played by a person.

In the beginning, the main research area of humanoid robots was related to how the robot moves and keeps its balance and little emphasis was given to its

upper-limbs, however over the last few years this trend has changed. The main focus of interest of humanoid robots has become the human-robot interaction, *"Research in humanoid robotics is Currently shifting from locomotion to interaction between humans and robots."* (Garcia et al., 2007). To make it possible and to extent the possibilities of this interaction, researchers began to investigate on how the upper-limb and more specifically the arms, can enhance this interaction in order to make it user-friendly and natural (Garcia et al., 2007). Rapidly it has become a strong area in which investment and evolution are notorious (Ross, 2005) (e.g. highly funded laboratories like Boston Dynamics are channeling their resources to this area). The latest advances have enabled the emergence of sophisticated and lightweight anthropomorphic robots, with some reasoning capabilities. It is strongly believed that the investigation in this area will not stop in the near future and humanoid robots will keep on evolving into becoming increasingly essential and useful elements in our everyday life. Taking into account the aging of the world population (e.g. in 2010, 25% of the Japanese population was over 65 years old (Tanie, 1999)), it is a common belief that in the future anthropomorphic robots will have a particularly high impact in the way to care elderly and handicapped people.

As such, anthropomorphic robots must be able to recognize the intentions of users and adjust their movements accordingly, using bimanual manipulation. It is important that all the tasks that are performed by the robot are executed in a smooth manner, pro-actively and physically identical to human actions. Therefore it can be foreseen the require to investigate new control methods and movement planning algorithms with the purpose to review reasoning and motion (Inoue, 1995). In short, the characteristics listed below must be regarded when looking for a humanoid robot that is able fully interact with humans in non-controlled environments:

- **Autonomous**: Able to cooperate with the user without the intervention of third parties (tele-operators), so the robot must react to external stimuli,

addressing situations with rationally triggered purposes;

- **Anthropomorphic**: Physical shape of the robot similar to the human-shape (requires two arms and two hands). Eases the manipulation of objects in environments cluttered with obstacles;
- **Bimanual**: Able to successfully and quickly perform tasks where the co-operative use of both arms is required to complete the task. Movements like re-grasp are only possible using two arms. It is also advantageous in human environments, giving a higher supportive capacity to the robot by enlarging its workspace;
- **Human-like movement**: Design the robot arm movements to be similar to human movements. This feature eases to predict the intention behind the movement and the will to cooperate with the user.

1.1.2 Why human like movements and human-like cognition in personal robots?

Anthropomorphic robots bring the main advantage of having physical characteristics (*human-like shape*) that allow the establishment of an easier partnership with humans. Taking this into account it can be said that these are the robots per excellence for social communication and for human-robot collaboration. However, the robot should not be too "humanized" so that the user feels easiness to start the interaction without the usual constraints and preliminary formalities that can be found between two humans who just met. On the other hand, the movements and behaviors should not be too robotized to keep credibility to the human's perspective. These human reactions were studied by a Japanese roboticist named Masahiro Mori (Chaminade et al., 2007). Mori concluded that despite the similarities between the robot and the human being evident at first glance, minor imperfections and flaws, such as resolution, speed or lack of facial expressions (Kobayashi et al., 2001) cause an effect of revulsion and denial in the

users. Mori illustrated this problem with his much famous and recognized "The Uncanny Valley" (see fig.1.2).

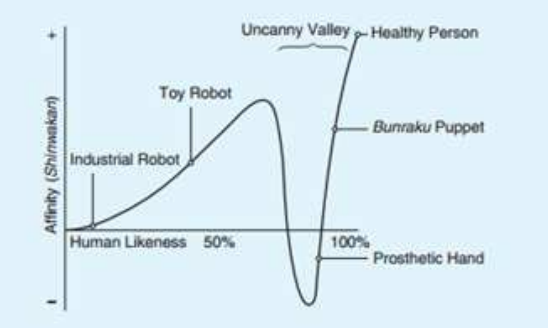


Figure 1.2: "The Uncanny Valley" by Masahiro Mori, (Chaminade et al., 2007).

Figure 1.2, shows the relationship between the human likeness of a robot and the generated affinity perceived by a human user. It was found that the reaction of the user to robots whose function is merely to interact with humans is very positive. This is mainly due to expectations that the human has in relation to the robot (Zhang et al., 2009). One should imagine a robot whose functions do not involve a direct interaction with the human user, for example, a robot that cleans and aspirates the floor such as the *iRobot*. These robots are easily integrated in the society because the user does not have expectations about their cooperative performance, it is only expected for vacuuming or cleaning a surface.

In robots whose goal is to establish a cooperative relationship with humans, small details such as lack of smoothness or the temporal coordination of movements cause rejection in perspective of the user. Although this type of robots are more advanced and adaptable to a vast range of tasks, they were in most cases rejected due to the expectation created by the user (Zhang et al., 2009) that arises from our habit to establish human-human interaction. The human user expects not only to be aided during the development of a task (e.g. transport a box) but also that the robot's movements, speed and the way the situation is processed to be identical to his own.

For a successful human-robot interaction the task should be completed by the

robot in an easy and understandable manner (Zhang et al., 2009). These features should be taken into account at the movement planning stage, and will turn possible the development of movements that are better accepted by the human-user who can recognize and understand them. When these goals are achieved, one reaches a new level of communication in which the human can prepare a response/reaction, even before the robots ends its movements, as if it was made by another person.

1.1.3 The starting point: Movement planning in the unimanual ARoS

Here it is going to be given a brief explanation on how the movement planning including how the anthropomorphic robot checks for collisions. *ARoS* was developed with the purpose of having a feasible unimanual movement planning, (see fig. 1.3).

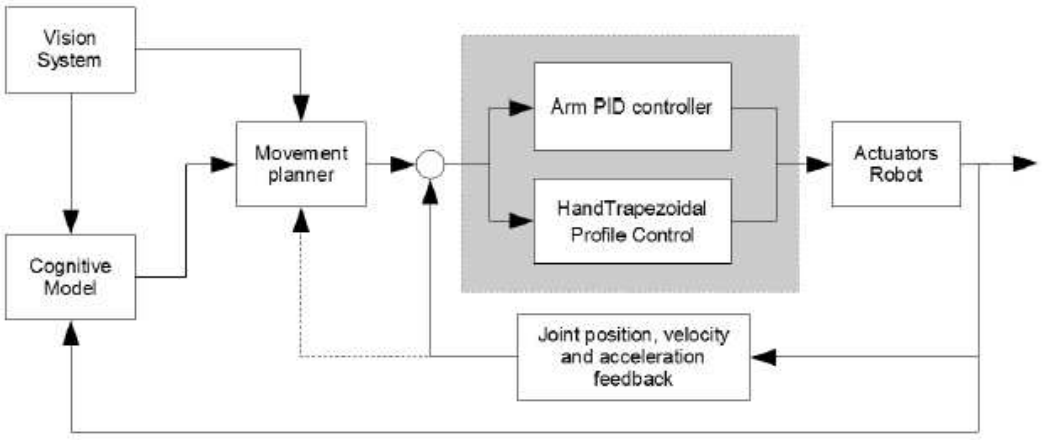


Figure 1.3: Schematic of the overall process for unimanual movements (Silva, 2011). The control of the arm and hand is made for each joint by built-in controllers.

There are several possible ways for a robot to grasp a given object with a previously specified grip type and a virtually infinite number of manipulator trajectories to grasp the object. Also note that the kinematic redundancy allows

the elbow to move in circles without any changes being visible in the end-effector (hand) position or orientation. There is also redundancy in the trajectory of the manipulator because there are plenty of possible paths with different timing requirements.

Accordingly, a movement planning was proposed to solve the existing redundancy problem at each level. Note that reaching, grasping and manipulating objects is based on some assumptions supported by human upper-limb observations:

- Movement planning is performed in joint-space (Desmurget and Prablanc, 1997);
- Synchronous movement of the arm joint angles (Breteler and Meulenbroek, 2006);
- Planning is separated in two different steps (Elsinger and Rosenbaum, 2003): a) Final posture selection (computed first); b) Trajectory selection;
- Final posture varies as a function of initial posture (Elsinger and Rosenbaum, 2003);
- Obstacle avoidance is achieved overlapping a direct movement from initial to final posture and a bounce movement from the initial to a bounce posture (Desmurget et al., 1998);
- Movements performed by the arm have a bell-shape angular velocity behavior (Wada and Kawato, 2004).

The planning begins by selecting a final posture considering the initial posture of the arm and hand, the feedback sent from the vision system (e.g. position, orientation) (see fig. 1.4). During the planning of a movement the trajectory (e.g. velocity, position) of each joint is computed. In a first instance the joint trajectory gives as output the direct movement from the initial to the end posture without taking into account obstacles and collisions. With the purpose to detect possible

collisions against obstacles placed in an intermediate position the planning system uses direct kinematics, to internaly simulate the movement from the beginning until the end while looking for collisions. If no collision is detected the movement may be executed but if the direct kinematics detects an obstacle it will calculate new feasible solutions until discovers a path in which no collisions were detected by the direct kinematics.

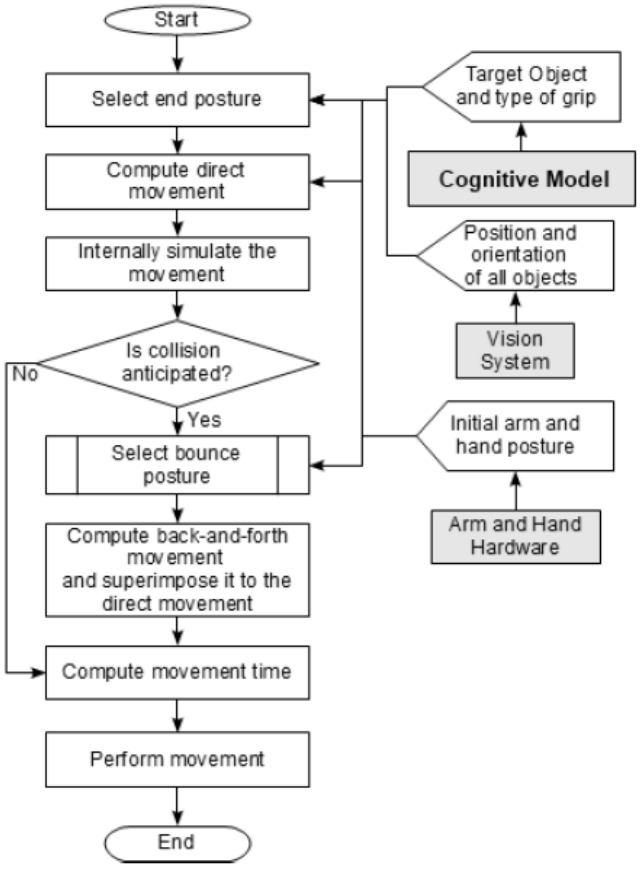


Figure 1.4: Schematic flow of processes to execute the movement planning used in **ARoS** to obtain unimanual collision-free movements, (Silva, 2011).

Digging more about achieving what was described above, the following methods were used:

- As can be read in literature, using a set of pre-programmed postures taken from practical data may solve the movement planning (Vaughan et al.,

2006). It can also be done through human demonstration, obtaining suitable postures;

- Admissible postures, obtained by the *Posture-based Motion Planning* model (Rosenbaum et al., 2001), may be set using inverse kinematic transformations to determine the joint angles of the arms positions and orientations for obstacle avoidance;

- *Matlab® Optimization Toolbox* extending the capabilities of Matlab® for optimization problems.

This process is subdivided in two steps, the selection of a series of finger joints and the selection of a series of arm joints (due to the trajectory selection).

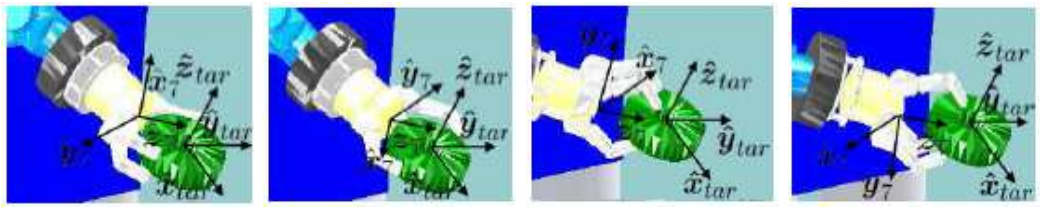
There are four different grip types (depending on the position and orientation of the finger joints) that may be considered, depending on the position of the thumb relatively to the object to be grasped:

- Side Grip thumb left: thumb on the left of the object (see fig. 1.5a);

- Side Grip thumb right: thumb on the right of the object (see fig. 1.5b);

- Side Grip thumb up: thumb is over the object (see fig. 1.5c);

- Side Grip thumb down: thumb is under the object (see fig. 1.5d).



(a) Side Grip left. (b) Side Grip right. (c) Side Grip thumb up. (d) Side Grip thumb down.

Figure 1.5: Grip Types.

A set of constraints on the vector of arm and hand joint angles were added in order to define the most suitable arm final posture. Having all the constraints set to determine a feasible posture, the optimal solution to reach an appropriate final posture for the arm is found.

It is worthwhile relevant to acknowledge that using inverse kinematics to calculate the bounce postures is a computationally demanding process and it becomes difficult to find an acceptable solution in a short time. Accordingly it is of great interest to find a faster method to calculate bounce postures and the final posture even in challenging scenarios cluttered with obstacles. With a faster method it would be possible to receive posture information in real-time and feasibly apply ARoS, in a joint task with a human.

1.1.4 Why should the robot movements be optimized?

Optimal motion planning problems for anthropomorphic manipulators are an active point of research in robotics through the last 20 years (Park, 2007). This is mainly because this kind of manipulators usually have a high number of DOF's in order to increase the capability to help during a shared task and the reachable workspace. Although being apparently a pragmatic goal, the optimization of a multi-joint robotic arm may be an ambiguous and difficult problem given its redundancy. That is mainly because there is a high number of different set of joint angles which can represent the same position of the end-effector on a given model (Toyoda and Fumihiko, 2004).

Various types of optimization methods have been developed to solve the optimal motion planning problem in a cluttered scenario. Within the context of this work it was needed to implement a non-linear optimization method once the mathematical models involved in this problems, based on postures, are not linear at its variables. A common non-linear optimization problem consists on minimizing an objective function $F(x)$ subject to a set of inequality constraints $g(x) \geq 0$, equality constraints $h(x) = 0$, and simple bounds p. With optimization

it is possible, ultimately, to implement smooth and human-like movements due to the minimization conducted.

There are three main requirements that must be obeyed while formulating non-linear optimization problems:

- **Human-like trajectories**: The optimized output values must result in a human-like trajectory. Since most of the times human movements are directed towards an object without deviations when there are no obstacles, the optimal solution obtained must follow the same behavior and be directed to the target, avoiding unnecessary and unbalanced joint movements. Note that this happens only when there is no obstacles on the way and if the human do not present any pathology that physically inhibit the human;

- **Quick calculus**: In order to keep the viability of the interaction and the human interested in the cooperation by sustaining his/her rhythm, the robot must calculate every movement in such speed that it won't delay the movement planning and by consequence the interaction;

- **Robust adaptation to changes in the scenario**: It is not of interest to have an optimization process adapted only to a pre-established sequence of movements, on the contrary it must give a optimal solution despite changes in the position of the objects in the scenario, presenting performance invariance despite the location of the obstacles and the target position.

Reaching an optimal solution of the arms movements with the above explained requirements, will make it possible to actively interact with the human partner in a fluent way. This process of optimization will also make it possible to move towards more complex movements and tasks since the time spent by the optimization solver will be much smaller, if correctly modeled and implemented, than the obtained in traditional inverse kinematics processes (see chapter 4, section 4.1).

1.1.5 Why bimanual manipulation? Which are the main challenges?

Considering the ability that humans have to plan and execute tasks involving grasping and manipulating objects, we see ourselves as the perfect resource of inspiration for running behavior studies with the main purpose to apply the recorded data to bimanual robots.

Anthropomorphic robots, like humans, must be able to render a fast bimanual motion planning, grasp and manipulate objects in a non-controlled environment. Although the task planning and synchronization of movements between both sides of the body is simple for healthy humans (Malabet et al., 2010), it is not an easy task to be accomplished for humanoid robots. Before performing an object manipulation, the following points must be taken into account (Vahrenkamp et al., 2011):

- The direct and inverse kinematics so that it is possible to know the end-effector position and the obstacles position. The first is with regard to the robotic arm joints. The second gives a feasible position of the arm joints knowing the aimed final posture of the end-effector. Note that for bimanual robots this process must be independently implemented for both arms;

- Geometry of the objects, so that it is possible to know where are the boundaries of the object and infer if it is possible to complete the proposed task for a specific object;

- Where the objects are in the workspace and simulate different motion paths. This requires the implementation a method to plan the motion of the manipulators.

There are several advantages resulting from the use of bimanual humanoid robots (see fig. 1.6), namely:

- Share and execute complex tasks that otherwise would not be possible. For example, one hand may keep the object fixed while the other performs a direct action on the same object;
- Using both arms, the robot is able to include a larger workspace, enhancing the capability to help humans without using torso or legs motion;
- Since the robot can perform bimanual activities the speed of execution may be increased, consequently decreasing the time needed to complete the task;
- Being able to execute bimanual activities that were goal orientated in *human-robot interaction*, considerably reduces the work that has to be performed by the human partner in joint tasks;
- With two arms the movements performed by the humanoid robot are better perceived by the human once it recalls his own, facilitating interaction in joint tasks.

Figure 1.6: Bimanual grasping trajectories made by ARMAR-III (Vahrenkamp et al., 2010), DLR humanoid Rollin' Justin (Zacharias et al., 2011), Sony's ASIMO performing a bimanual activity, Qrio playing violin, Robonaut-2 performing a bimanual activity (Diftler et al., 2012).

Human environments might be challenging, uncontrolled, dynamic and hard to perceive reliably but it is believed that the use of two robotic arms may help cope with these challenges. Using bimanual manipulation, it is possible to control two objects, control the grasped objects with respect to one another (Edsinger and Kemp, 2008a), interact with the user at the same time and even accomplish

complex tasks like re-grasping. Accordingly, and keeping in mind all that may outcome from a bimanual solution, it is of interest to pass from a unimanual to a bimanual solution.

So why there are no humanoid robots performing autonomous bimanual and human-like object manipulation the way humans do? One of the major reasons is due to the fact that humanoids with bimanual capabilities are a complex challenge. Some of the most relevant difficulties are now explained:

- In robotics, controlling a large number of DOF's in a willful manner (usually humanoid manipulators have a large number of DOF's) is computationally expensive. It is important to note that if a robot grasps and manipulates an object, the object also adds extra DOF's, requiring even more computational resources;

- In terms of object manipulation, it is already known that objects add extra DOF's, but if the object does not have a rigid structure (e.g. towel) even more DOF's are added. These kind of objects are also a source of unpredictability because it is difficult to understand exactly where are its limits. Another source of unpredictability may be the flexibility of the object, how is it going to react to manipulators and how much strength should be applied;

- When cooperating with human users in real time situations, chaotic contexts may be hard to filter and to viably perceive all the incoming visual information, once it is required to choose between a very large range of possibilities. It is important to understand that the decision made for how to perceive the visual information must be coordinated with the movement of both arms;

- One should be aware that the above explained problems are very much enhanced if the movements have to be human-like. This is considered another goal, it is difficult to accomplish, specially in unpredictable situations. It

may be considered as another constraint, restricting and avoiding possible but not human-like configurations of the arm joints.

1.1.6 Which kind of bimanual movements does the human being usually perform?

During the day an active person uses its upper limbs to grab and manipulate objects several times. Such movements are performed in a natural manner in the majority of times, therefore there is no need of a conscious thinking (Janssen et al., 2008). Even the simpler movements implicate the coordinated use of an elevated number of muscles acting on the tendons and the joints. However, in order to execute these movements properly, it is necessary to plan them in the Central Nervous System (CNS). This planning is designed in a careful way, especially if it involves the two upper limbs (bimanual movements).

As the number of different bimanual tasks that are possible for humans to perform is extremely high, it is surprising that a human being is able to properly and easily select a posture without spending time and reasoning (Weigelt et al., 2006).

Furthermore it is relevant that all the tasks accomplished in a bimanual way are contained in a set of sub-tasks. Knowing this it is specially important to understand and to create a theoretical framework of the bimanual behavior (Guiard, 1987) (see fig. 1.7) on bimanual robots, which is now presented:

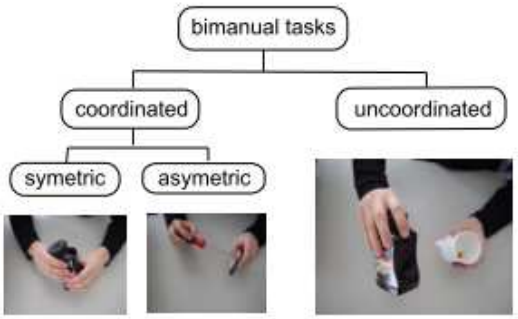


Figure 1.7: Classification of bimanual tasks (Zollner et al., 2004).

- **Coordinated two-arm manipulations**- Consists of having both hands working synchronously on a given task. Note that those tasks can be subdivided into two categories:

    1. **Symmetrically coordinated tasks**- Includes movements in which both hands play the same role (e.g. transporting a cup of tea), (see fig. 1.8a);

    2. **Asymmetrically coordinated tasks**- Includes movements in which the roles of each hand are different from one to another (see fig. 1.8b).

- **Uncoordinated two-arm manipulations**- Happens when there are tasks that do not need to be performed with coordination between both arms. This specific bimanual behavior allows each hand to be mapped for a unimanual agent, albeit in some cases the coordination between both arms would be advantageous to avoid several temporal and geometrical problems (see fig. 1.8c).

(a) (b) (c)

Figure 1.8: Accordingly to (Guiard, 1987), the figures presented above show his view on the different types of bimanual tasks.

In 2011, Zacharias et al. (see (Zacharias et al., 2011)) proposed to differentiate and diversify the work started by Guiard (see(Guiard, 1987)). It was intended to visualize the problem of bimanual manipulation giving a special emphasis to logical and geometric relations. Accordingly it was suggested the following:

- **Logically independent**- When grasping, placing or moving two objects with different hands. It may be compared to two unimanual tasks happening simultaneously. Note that the relative position of both manipulators related to each other must be kept under care;

- **Logically dependent**- When a well defined sequence of movements using both arms, must be obeyed. Logically dependent tasks can be subdivided on the following:

    1. **Non-cooperative**- Such kind of bimanual tasks require that a set of subgoals were achieved in a defined order without the need of cooperation between both arms. In fact, it is easy to infer that, these kind of bimanual activities have no need of both arms to successfully complete the task and it is only needed that the subgoals were sequentially achieved. Despite this, a coordinated bimanual execution speed-up the task (e.g. one hand opens the microwave while the other simultaneously carries pop-corn into to the microwave);
    2. **Cooperative**- Happens when a given bimanual task has to be coordinated to be successfully completed. It is important to keep in mind that these tasks imply a temporal and functional interrelation. A classic example for this logically dependent cooperative task is to open a bottle, while one hand removes the stopper the other holds and stabilizes the bottle.

- **Spatially independent**- On a spatial point of view, the task performed by one manipulator do not restrict the workspace and the actions of the other manipulator. There is no spatial influence between each manipulator and the tasks may be executed without spatial restrictions;

- **Spatially dependent**- On a spatial point of view, the task performed by one manipulator restricts and influences the movement of the other manipulator. It is important to ensure that there are algorithms to avoid

collisions between both manipulators and the objects being carried during the movement.

Note that there are some absolute relations between the above explained classifications. If there is any logically dependent cooperative task, that task is surely going to be spatially dependent.

1.1.7 From Unimanual to Bimanual movements. Which were the key changes?

There is the need that in the future, **ARoS** owns bimanual capabilities to successfully share any cooperative task with the human user. At the moment our main purpose in bimanual manipulation is to have the ability to perform Asymetrically Coordinated and Logically Dependent Cooperative tasks (see section 1.1.6). Considering this and starting from an unimanual platform (see (Silva, 2011)), several adaptations and additions must be conducted. First of all, considering the hardware layer, it was used a second arm called *Light-Weight Arm 7 DOF* (see fig.1.9) from *AMTEC Robotics/Schunk*®, a second robotic hand was also needed with the purpose to grasp with the left arm, it was decided to use a *BarrettHand-BH8-series*.

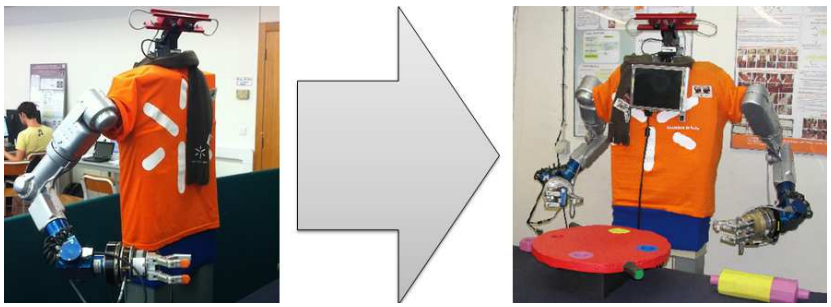


Figure 1.9: Previous version of **ARoS** at the left side of the figure. Current Bimanual version of **ARoS'** at the right side of the figure.

Note that the referenced arm (see fig:1.9) was placed on the left side of the anthropomorphic robot (*ARoS*) and its right side has an arm and robotic hand

with the same references. The Light-Weight Arm 7 DOF was designed to be on the right side of an anthropomorphic solution see (Silva, 2011). Therefore, some adaptations had to be conducted in order to keep the same mirrored behavior, as it will be explained later in section 3.4.

In the software layer there must be taken into account some adjustments to simulate left arm movements and to see it properly performing movements. It is also necessary to keep in mind that there are several implications resulting from having a two-arm robot, like the following:

- A new simulation environment with the new arm and its control-panel had to be developed in order to feasibly recreate not just unimanual but also bimanual simulations.
- The Check-Collision algorithms and its constraints must be enabled to both arms;
- Each arm has now to avoid collisions with the Anthropomorphic body;
- Direct and Inverse-kinematics must be developed to adapt to the global position and orientation of the left arm, accordingly to this it is not a viable solution to use the kinematics of the right arm;
- The non-linear optmization problems must be formulated for unimanual movements with slight differences in matrix transformations between the left and the right arms.

1.2 Contributions of this Dissertation

The aim of this MSc. project is to endow an anthropomorphic robotic platform with unimanual and bimanual movements, firstly by adapting the left arm kinematics and then implementing an optimized movement planning for real-time and *human-like* movements for both arms.

This research project was developed at the Mobile and Anthropomorphic Robotics Laboratory (MARLab) of the Department of Industrial Electronics and the Algoritmi Center at the University of Minho (Campus Azurém). The anthropomorphic robot **ARoS** was used to develop this project. **ARoS** is composed of a static torso and equipped with two arms with 7 DOF's (shoulder - 3 DOF's, elbow - 1 DOF, wrist - 3 DOF's), each one with a 4 DOF's robotic hand (with three fingers), it also has a stereo vision system mounted. This project is intended to continue and expand the work started by Silva (2011).

The human robot interaction during the construction a task is not yet a fully solved problem. This is mainly due to the complexity associated to object manipulation. During unimanual movements there are redundant degrees of freedom of the arm and hand which allows a task to be solved in many different and possible ways. This complexity is exacerbated in the presence of numerous obstacles in the workspace. Optimization processes are required in order to achieve, fluent, smooth and collision free movements for real-time interaction. It is also needed that the movements maintain human-like features in order the ease the interaction for the human. Finally, these features need to be adapted for bimanual movements, because these presents important advantages over unimanual movements (e.g. increased workspace, concluding more complex tasks).

This dissertation project was integrated under the following research projects:

- European IP project "JAST - Joint-Action Science and Technology financed by the European Commission" (ref. IST-2-003747-IP);
- European project PF7 Marie Curie "NETT - Neural Engineering Transformative Technologies".

1.3 Dissertation outline

The remainder of this work of dissertation is organized as its shown:

Chapter 2 presents the state-of-the-art in unimanual and bimanual manipulation and the methods used in these robotic platforms;

Chapter 3 presents a overall view and fundamental aspects of the **ARoS'** robotic arm, also describing a theoretical implementation of direct (forward) and inverse kinematics for the left arm;

Chapter 4 presents the implementation of a non-linear optimization used to solve the posture based problem;

Chapter 5 presents the physical structures used to implement and test the developed movement planning system;

Chapter 6 presents the validation of the developed work for a construction sequence of the a Toy Vehicle, describing its main features and characteristics;

Chapter 7 presents the validation of the developed work for a construction sequence of the a Space station scenario, also describing its main features and characteristics;

Chapter 8 presents a summary of the collected results, withdrawn conclusions and a discussion to related work.

Chapter 2

State-of-the-Art

Having in mind the main goal of a human-robot interaction, it is required to translate into movements the decisions and choices that are previously taken. It is also necessary that these movements are collision free, fluent, smooth and must allow the human interpretation of the intention behind. Here is shown how motion planning of anthropomorphic robots directed to human-robot interaction is being currently developed, giving a special attention to the object manipulation. Firstly it will be overviewed and examined unimanual planning methods and later bimanual ones.

2.1 Unimanual Movement Planning

Within the unimanual manipulation tasks it is of interest to check the difference between local and global methods related to collision-free path planning. Global methods usually act in two stages (Kondo, 1991), the first stage is generally done off-line and consists of making a representation of the free configuration space, (CSFree). With that objective in mind there are several ways to represent the CSFree: the octree, the Voronoï diagram, the grid discretization and probabilistic road-maps. After choosing a way to represent the CSFree it is used a method in order to construct the free configuration space which uses the representation of

the CSFree (LaValle, 2006). This method includes approaches where the planning is performed by having a absolute coordinated system looking for collision-free paths during the whole movement. One must keep in mind that the obstacles are mapped in the system configuration space as a forbidden area (Latombe, 1999). The local methods use on-line algorithms to check for collisions during motion and are usually used in real-time applications prompting reactions and avoiding obstacles by triggering fitting algorithms whenever it is needed, even in the middle of a movement.

2.1.1 Global Methods

2.1.1.1 Sampling Based Motion Planning

An usual approach to solve the collision-free path through global methods is sampling-based motion planning (SBMP) (see (LaValle, 2006), chapter 5). The idea beneath is to avoid the explicit construction of C_{obs} (obstacle configuration space), and instead conduct a search that probes the C-space (configuration space) with a sampling scheme that continuously checks for collisions, as may be seen on the figure 2.1 The problem related to *Sampling-Based Motion Planning* algorithms is that there is no guarantee if the problem will be solved. Even if it is, the solution obtained probably will not be the optimal solution.

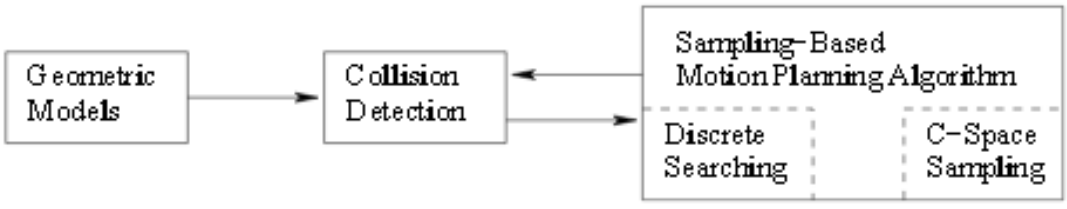


Figure 2.1: Sampling-Based Motion Planning, uses collision detection separately from the particular kinematic models, (LaValle, 2006).

2.1.1.2 Rapidly-exploring Random Trees

Over the last years, several randomized data structure for path planning, approaches have been successfully applied to the path planning problem. One of the most famous approach consists in Rapidly-exploring Random Trees-RRT (LaValle, 1998), (see fig. 2.2).

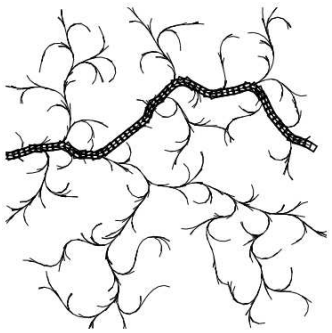


Figure 2.2: A 2D projection of the RRT method, (LaValle, 1998).

This approach has proven to return good performance in high-dimensional spaces even if algebraic and differential constraints are used. The main optimization of this method related to previous randomized data structures is about the use of as less arbitrary parameters as possible, leading to easier performance analysis and behavioral consistence (see, (LaValle, 1998)). In short its operation consists of a search in a metric space, (X), for a continuous path from an initial state, x_{init} to a goal state x_{goal}, connecting start and goal configurations (LaValle, 1998, 2006).

An improvement applied in the base of the RRT method is shown in (Kuffner Jr and LaValle, 2000), it explains a method that works by incrementally building two Rapidly-exploring Random Trees (RRT's), in which one of them is rooted to the start configuration and the other to the goal configuration, (see fig. 2.3).

Each tree explores the space around them while advancing towards each other through the use of a simple greedy heuristic. This approach has some advantages, like not requiring parameter tuning and pre-processing. Some practical tests were conducted, one of them was related to the generation of collision-free motions for

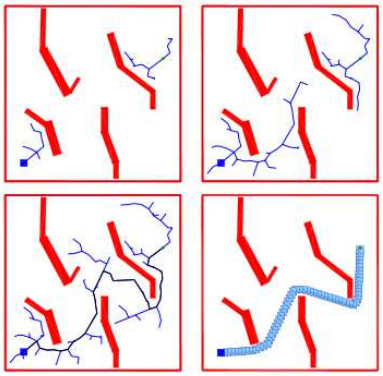


Figure 2.3: Two RRT's growing towards each other, (Kuffner Jr and LaValle, 2000).

rigid objects, it was also conducted a test consisting on the generation of collision free motion planning for a 6-DOF PUMA manipulator (see fig. 2.4).

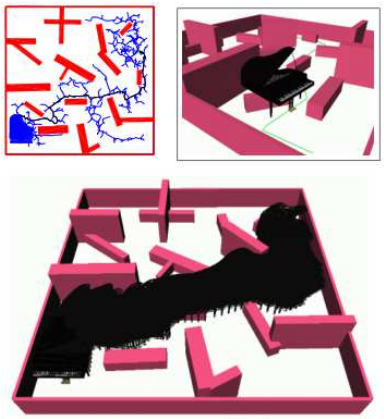


Figure 2.4: Path planning using RRT's to move the piano, (Kuffner Jr and LaValle, 2000).

The task carried out by the 6-DOF PUMA manipulator consisted of transporting a piano from one room to another, which took about 17 seconds to be planed. One may conclude that the length of each tree needs to be optimized and an approximation to the nearest-neighborhood of the RRT is required to reduce computation times (Kuffner Jr and LaValle, 2000).

2.1.1.3 Derivations from RRT method - *Tangent Space Sampling* and *First-Order Retraction*

In recent years it was explored and submitted a new kind of globally randomized joint space path planning for robotic manipulators that has several constraints in its workspace. Mike Stilman presented two random sampling methods of joint configuration for articulated manipulators (*Tangent Space Sampling* (TS) and *First-Order Retraction* (FR)), that were related to task space constraints and exploration of joint redundancy. This means that in addition to find a collision-free joint-space path it is needed to satisfy workspace constraints for the end-effector using a non-predetermined path. It is necessary to explore and improve these algorithms because of the real world evolving needs (e.g. opening doors [see fig. 2.5], pushing carts or removing rubble), once the robot must have the required dexterity to accomplish the objective, exploring joint space paths that satisfy the constraints. The proposed methods (see, (Stilman, 2006, 2010)) were derivations from the basic RRT-algorithm.

Figure 2.5: Simulation of the movement planning to open a door using Stilman's et al., RRT method. Note that the robot most avoid joint limits and the box placed in random positions to successfully complete the task (Stilman, 2010).

Note that the resultant movement is not human-like and its unpredictability is not acceptable to be considered its use in humanoid robots. Even though it is also relevant to note that these methods are faster and less variant to different problems then some predecessor RRT algorithms. The obstacle avoidance has a huge non-linearity related to joint configurations leading to the necessity of use

local optimization. Testing shows that the output is not yet ideal when there is the need to avoid obstacles. It is also important to keep in mind that the unnatural path may be the result of an weird start or a awkward goal configuration or even due to the probabilistic path planning currently used.

2.1.1.4 Ergonomics criterion

Zacharias et al. (2011), proposes to find a human-like goal configuration using an ergonomics research criterion to seek and select the arm initial and final configurations. Much emphasis is given to this once the selection of an arm's target configuration strongly affects planning time and how natural a path looks. Restricting the motion planning to regions inside the boundaries of the manipulator workspace where human-like manipulation is more probably achieved brings great advantages related to the time spent calculating the path and how natural it looks in a geometrical dimension. The above concepts were evaluated using the right arm of the humanoid robot *Rollin' Justin*, (see fig.2.6 and (Zacharias et al., 2011)).

Figure 2.6: Simulation of the robot Rollin' Justin used to test human-like geometry planning (Zacharias et al., 2011).

It is worthwhile noting that the above explained Zacharias et al., method is only focused on the geometry of the motion planning and no significance is given to temporal results (e.g. time accelerating) of the movements. Accordingly to (Rosenbaum et al., 1999; Lommertzen et al., 2009), it is of outmost importance to consider that the speed and acceleration with which the movement is performed

also have a relevant impact in the human-likeness sensed by the human user. To keep the temporal profile under care it is proposed to apply the minimum jerk principle to the joints of the arm and hand, implementing a *Bell-shaped* acceleration and velocity profile, resulting in a smooth movement in joints space (Lommertzen et al., 2009).

2.1.1.5 Movement planning using optimization methods

It is important to acknowledge that there are some movement planning approaches like those using optimization methods. Usually these approaches requires that the obstacles and the joint limits were defined as constraints.The most relevant ones may be considered the *Optimal Control Theory* (see, (Galicki, 1998)) which is usually used to solve equations for the motions of at most 3-link manipulators due to theirs non-linearity. The *Non-Linear programming* (see, (Park, 2007)) tries to minimize the calculus time of the motion by minimizing the overload trajectory, consequently the resultant times of motion planning are considerably reduced, therefore being considered a good method to be implemented in 6 or more DOF manipulators. *Tessellation of Joint* (see (Ozaki and Lin, 1996)), requiring less CPU time but may accumulate numerical errors in small time-intervals lowering the results accuracy. Configuration space (see, (Shiller and Dubowsky, 1991)), which assures stable convergence but this refinement of tessellation consumes lots of resources increasing exponentially CPU times, becoming impossible in real-time cooperative tasks. *Dynamic programming*, (see (Fiorini and Shiller, 1996)) consists in two main steps: the computation of the trajectory and its refinement with dynamic optimization. Note that all of these methods address the movement planning problem by generating both path and speed profiles which lead towards a more human-like movement. Unfortunately in most cases the end-effector is restricted to obey pre-programmed paths, making it impossible in real-time situations.

Accordingly, one may say that most of global methods even when used for

unimanual planning are considered a computationally demanding challenge due to their impracticability and difficulty to represent high-dimensional configuration spaces (Kuffner et al., 2005) also note that there are some exceptions to this statement. Another drawback is related to geometric aspects causing difficulties to find an optimal solution.

2.1.2 Local methods

2.1.2.1 Potential field method

The *Potential Field Method* proposed by Khatib (1985), is widely recognized as the first relevant local method. Assuming that the robot evolves in a potential field in which the sum of an attractive potential represents the desired goal position and repulsive potentials represent obstacles and joint limits of the arm. The main focus of this methods is to path planning but usually the temporal description of the path is not taken into account, rendering a less natural path.

2.1.2.2 Attractor dynamics

Iossifidis and Schoner (2006) addressed the attractor dynamics approach by generating a goal-directed movement capable of react on the sudden appearance of an obstacle also respecting joint limitations (see fig. 2.7). The robot movements consisted of straight-line tracks (up and down) by using two additional direction angles acting on the end-effector.

To avoid collisions and to respect joint limits in a scenario full of obstacles the elbow may make use of its redundancy lifting or lowering, keeping the same movement and hand orientation (in this case the movement is a straight-line). Note that the forearm rotations around longitudinal axis were not taken into account and the movements were pre-programmed.

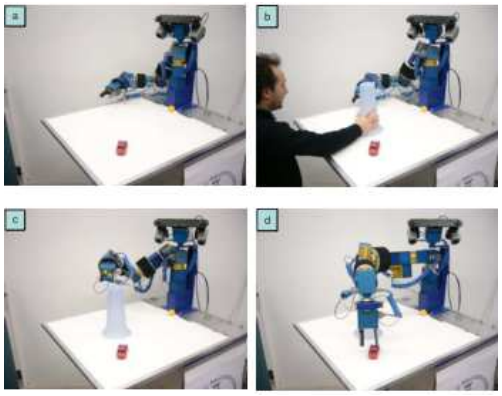


Figure 2.7: Manipulator evading dynamic obstacles using attractor dynamics methodology (Iossifidis and Schoner, 2006).

2.1.2.3 Extension to basic Attractor Dynamics method

In 2010, an approach previously developed in (Iossifidis and Schoner, 2006) was extended by Reimann et al. (2010). The main contribute is related to the real-time avoidance of several arbitrarily placed obstacles near the joints of a robotic arm. The feasibility of this approach was verified by implementation on the 8 DOF's robotic arm CoRA. The performance was characterized by detecting failures of the motion planning when varying (increasing) the number of obstacles. Accordingly, this method is capable of successfully find paths in a large range of situations (Reimann et al., 2010). Although presenting good results, the singularities and collision of the arm against himself were not considered. Additionally the sudden movements generated when avoiding obstacles calls into question the Human-robot interaction once the prediction of the robotic arm movement in this kind of situations becomes barely possible for the human user.

2.1.2.4 Dynamical Movement Primitives

A different approach to movement planning using dynamical systems is the Dynamical Movement Primitives (DMP) based on a group of nonlinear differential equations with a well defined attractor dynamics (Ijspeert et al., 2002). The consequent movement planning is autonomous, adapted to external perturbations

and is possible to change his input values by changing his perceptual variables. It is important to note that the attractor dynamics landscape learns from a demonstration given by a human teacher. The main purpose of this method is to be used in humanoid robots making possible to accurately encode concrete trajectories and being able to faithfully and easily replay them for different situations and goals. This was tested and evaluated with a humanoid robot with some results (Ijspeert et al., 2002) suggesting that a learned movement can be easily reused to generate robust paths towards different goals.

2.1.2.5 Extension of dynamical movement primitives

Later, Hoffmann et al. (2009) presented an improvement over the original framework presented by Ijspeert et al. (2002). Following this work, the movement planning can be generated taking into account singularities and without quick accelerations. The formalisms related to obstacle avoidance were also improved by exploiting the strength and behavioral invariance against perturbations with the use of differential equations to make the robot's end-effector steer around an obstacle. Some practical tests were conducted by maneuvering a cup to different target positions preventing the end-effector to collide with a moving ball (see fig: 2.8).

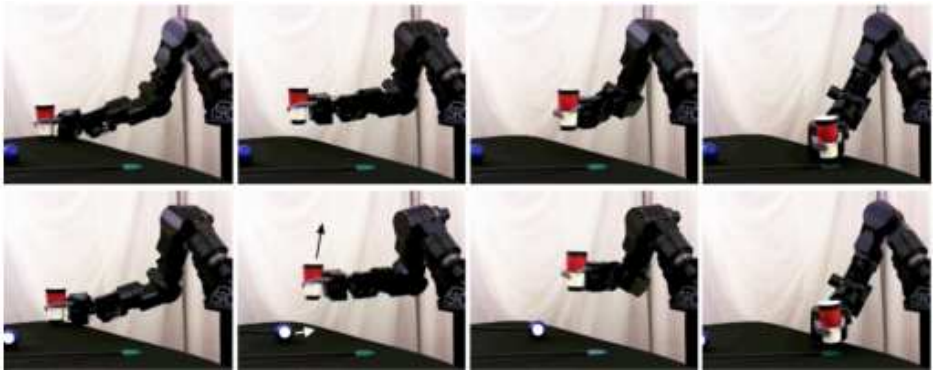


Figure 2.8: First row of movements show motion without obstacle. The second shows its behaviour with a movable ball on the way of the end-effector (Hoffmann et al., 2009).

It is worthwhile noting that the newly added equations were inspired in human behavioral experiments and biological data, resulting in a more human-like motion, (Hoffmann et al., 2009). Nevertheless numerical problems arise when the target positions were changed.

Pastor et al. (2009) presented his extension to Ijspeert et al., original DMP framework, using dynamic equations. The purpose of his work was to provide an approach for learning motor skills from human demonstration using non-linear differential equations to learn and afterwords reproduce the movement. To test the feasibility of this approach tests have been conducted on the Sarcos robotic arm, that has learned a pick-and-place operation and a water serving task from human demonstration. Taking into account the results obtained one may say that this framework allows object manipulation with good naturality, real-time obstacle avoidance with objects in the scenario and invariant behavior to new defined goals (Pastor et al., 2009). Even though this framework has presented good results, the joint limits and arm joints collision avoidance (with the own arm) were not taken into account.

2.1.2.6 Dynamic Potential field

In 2008, Park et al. (2008) extended the work described by Ijspeert et al. (2002) with the purpose to represent random movements in the manipulator space (see fig. 2.9).

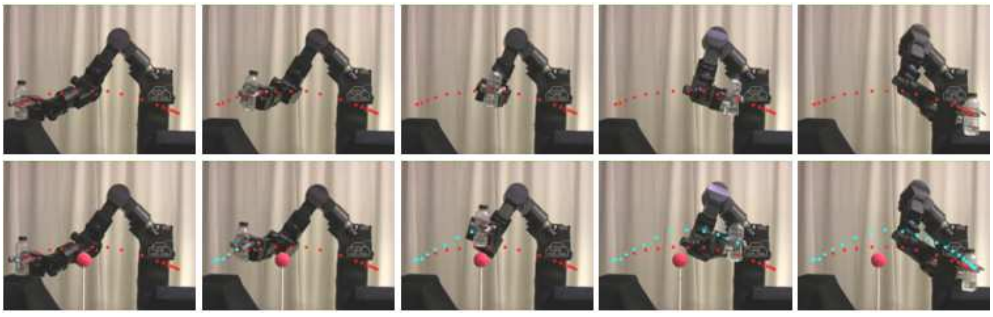


Figure 2.9: Manipulator escaping from moving obstacles using dynamic potential fields (Park et al., 2008).

Further motion equations were added to avoid obstacles using gradient of a potential field focusing the area to avoid. Demonstrations using a anthropomorphic robot were was conducted. Accordingly, during its motion the robotic arm was able to smoothly avoid an obstacle using dynamical potential fields.

2.1.2.7 Generalization of discrete dynamic movement primitives

Ude et al. (2010) presented a way to enable the generalization of sensory motor knowledge by imitation. The new activities use *Gaussian regression*, in which the target and other characteristics of the action (e.g. cartesian positions) were used as queries to create a proper control policy. One should also take into account that this model uses Non-linear Dynamical systems with the purpose of motor representation. It is also shown that this methodology may be used with a 3D vision system of a humanoid robot. Considering the experiments conducted, one may say that the proposed approach can be used in a feedback loop by live modification of a Dynamic Movement Primitive (DMP). It is relevant to verify that there are tasks in which the movements do not have a smooth transition. As an example the author states that movements with obstacle avoidance from different sides (at least two), this should not be applied simultaneously due to collisions risk. This indicates that this approach is usable but there is a need to add complementary clustering procedures that define suitable sets of trajectories.

2.2 Bimanual Movement Planning

In the previous section (see section 2.1) it has been summed up some of the most used and developed methods using Adaptive Computing System (ACS) applied to unimanual anthropomorphic robots, nonetheless, despite the method, the set of tasks that can be accomplished are very limited. Accordingly to this it is of great interest to have an anthropomorphic robot with bimanual capabilities.

The ability to grasp objects with both hands enables anthropomorphic robot

systems to fully employ the human behavior in human-centered environments (Vahrenkamp et al., 2011). Next it will be evaluated the state-of-the-art in bimanual activities.

2.2.1 Probabilistic RRT-based

Vahrenkamp et al. (2009) presented a computationally efficient solution for planning of bimanual motions with manipulation and re-grasping tasks, solving the inverse kinematics problem by combining it with a probabilistic RRT-based motion planning approach. This strategy resulted in a feasible inverse kinematics solution with low computational cost, without needing fallible iterative methods. Accordingly, the above explained method was successfully tested in the humanoid robot ARMAR-III which performed dual-arm tasks in a kitchen environment. This work was later improved by integrating a planner for collision-free bimanual grasping motion, which means that beyond finding a feasible grasp it also look for a collision-free one for each arm, (Vahrenkamp et al., 2010). This planner combines three main tasks required to grasp an object: finding a potential grasp, solve the inverse kinematics and at the same time search for a collision-free track that brings the hand for the final posture required to grasp the object. It is also presented a bimanual planner that generates grasping motions (see fig. 2.10).

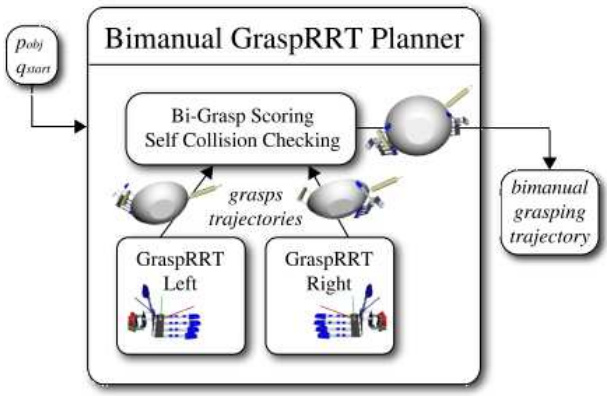


Figure 2.10: Bimanual Grasp-RRT movement planner (Overview), (Vahrenkamp et al., 2010).

Accordingly to the author, in this method two separate RRT (one for each hand) were started at the same time. Accordingly to the author and going more specifically onto the used bimanual method it can be said that it is set to search and save grasps until the main thread runs out of time.

Once ended, the thread collects the viable grasps and its trajectories separately for the right and the left arm, trying to find an acceptable bimanual solution. The feasible solutions are built and scored, if that score is above a certain value the planner check both grasping paths for potential self-collisions. If no collision was detected the solution for both arms is saved together with the grasping information, (Vahrenkamp et al., 2010). Following (Vahrenkamp et al., 2010, 2009), the results obtained were good but this method is not adapted to dynamical environments.

2.2.2 Behavior-based approaches

It should be noted that dynamic tasks remain a problem to be solved since there are no systems that have good performances under sudden perturbations. With the aim to tackle this problem, researchers tried to implement a behavior-based approach to bimanual manipulation. This approach was tested in a humanoid robot to support a person placing objects in a shelf (Edsinger and Kemp, 2008b). Accordingly there are some pre-programmed models of behavior that can be switched depending on the user needs, providing a better adapted and specific feedback to the situation in which the Human-robot interaction is occurring. Despite the advantages felt in the interaction between the Human and the robot, the weight and the complexity due to the implementation of algorithms for movement in bimanual manipulation is considerably high.

A transverse disadvantage of both probabilistic and sampling-based methods is the same felt when applied to unimanual movement planning and is related to the geometrical paths that may be unpredictable and doubtful. As it is seen by Vahrenkamp et al. (2011), the problem may be solved and the movement planning

may find a solution but it is possible that the operation is not comprehensible for the human observer. Here, Zacharias et al. (2011), focus on the geometry of the paths but disregarded temporal (velocity and acceleration) human-likeness of the motion. It is known that is important to consider velocity profiles of both arms and their interdependencies.

2.2.3 Learning by Imitation

Accordingly to what has been stated it is important to find an approach that addresses and solves the problem above stated (see section 2.2.2). It is essential to have a movement planning that takes into account both temporal and spatial coordination. It is being referred in literature that one possible way is the Learning by Imitation method using Programming by Demonstration (PbD) (see fig. 2.11), (Zollner et al., 2004; Asfour et al., 2008; Gribovskaya and Billard, 2008).

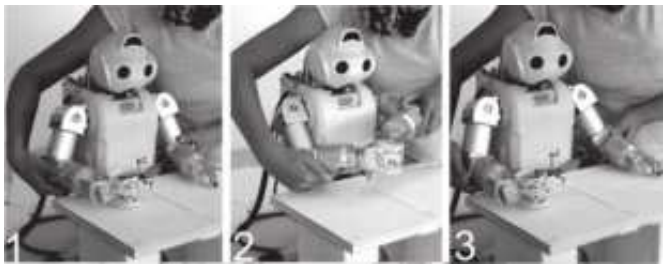


Figure 2.11: Three steps of a Learning System using Programming by Demonstration, (Gribovskaya and Billard, 2008).

Specifically regarding the method described by Gribovskaya and Billard (2008) consists of a *learning system* that processes data gathered during the demonstration of the task to extract constraints, and a motor system that give as output dynamical movements while fulfilling all the coordination constraints given by the learning system. The author assumed that, for discrete movements the position of both end-effectors in relation to each other remains a viable variable to capture the coordination constraints. During the learning process key postures related to the most difficult and constrained positions of both arms were

extracted. Once this method works cooperatively with non-linear dynamical systems, the extracted data is converted into attractors (stable fixed points). This approach has two really strong advantages:

1. The rendered behavior has a naturally triggered action against perturbations and the attractors may be set over several key points;

2. The motion becomes invariant to rotations of the humanoid robot once the variable that collects the coordination constraints of both arms represents just the relative position of both manipulators, it does not consider the rest of the body making it irrelevant at this point.

2.2.4 Programming by demonstration

Programming by demonstration (PbD) (see fig. 2.12), is a *Learning by Imitation* method used in order to observe, learn and execute, in this case, bimanual tasks previously performed by the human user. Here it is specially important to extract as much information as possible before starting to execute bimanual tasks (Zollner et al., 2004).

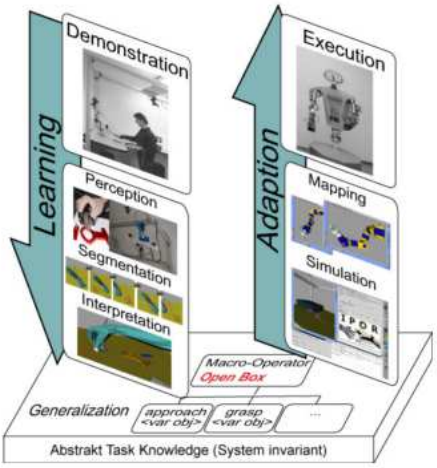


Figure 2.12: Programming by Demonstration Cycle from a Human Demonstration to a Robotic Target System, (Zollner et al., 2004).

After extracting information the trajectories were adapted and transferred to the robot. The proposed method was validated with the task of opening a jar. The coordination of bimanual tasks is based on the *Petri net* model. This model is being widely used to model dynamical systems and within the context of (Zollner et al., 2004), is used to efficiently represent control and data within only one formalism. This approach allows the structure of the task to be kept and successfully accomplished, but fails in flexibility and adaption to new scenarios.

2.2.5 Hidden Markov model

Asfour et al. (2008) presented a *Hidden Markov Model* (HMM) related to imitation learning. Continuous hidden Markov models are used to generalize movements demonstrated to a given robot multiple times, this is accomplished using key points that are identified as common to all demonstrations. It is also shown that HMM may be used to detect temporal dependencies between both arms in bimanual tasks. Experiments using the explained method have been performed using a kinematics model of the human arm and being able to reproduce the arm movements with human-likeness. The method does not take in account translations or rotations of the robot which can hinder the movement planning.

2.2.6 Passive Motion Paradigm

During 2009 it was proposed another way of generating coordinated bimanual movements using dynamical systems. The method consists of an extension of a motion planning with an artificial potential field approach (*Passive Motion Paradigm*-PMP, (Ivaldi et al., 1988)) which is based in attractor dynamics (based on, (Tsuji et al., 1995)) and has also been applied to robot reasoning (based on, (Mohan and Morasso, 2007)). The attractor dynamics makes it possible to have a point attractor in a desired target and other task constraints applied to the end-effector and then mapped into the corresponding force field inside the joint space. The repulsive force fields are used to implement joint-limit avoidance, repealing

joints from singularities and other limits. At the same time, it is also possible to focus on trajectories with the help of external constraints. The PMP model is composed of different virtual force fields having a cognitive meaning, allowing an easy and natural integration of planning with reasoning. The practical feedback of this method is promising but all the simulations were conducted in simple scenarios and performing simple bimanual activities. A great advantage of this method is a native characteristic of the attractor dynamics, in which is possible to control movement timings and velocity during the execution of tasks leading to natural and human like movements.

Reviewing the state-of-the-art of unimanual and bimanual movement planning, becomes easy to understand the need to develop a planning algorithm capable of planning movements with human-likeness in environments cluttered of obstacles. It is also important that this method may work in real time so the cooperation between human users and robots becomes possible and much easier for the human.

Chapter 3

Kinematics of the unimanual and bimanual ARoS

In order to fully understand the concepts presented in this section the robotic arms used will be presented first, since the kinematics depends on characteristics like number of DOF's and length of joints which are associated to the arms. Later on this chapter information is given about joint limits in both arms and how the **ARoS'** arms work. Afterwards theoretical information about the direct and inverse kinematics is presented, and then an explanation is described concerning how the direct kinematics is calculated for the left and right arms. Finally an explanation is presented about how the inverse kinematics of each arm was implemented.

3.1 Fundamentals of direct and inverse kinematics

Here will be presented details about direct and inverse kinematics and some notation that will be useful in order to better understand sections 3.3.1, 3.3.2, 3.4.1 and 3.4.2.

Usually, **kinematics** refers to the study of geometrical and temporal based

properties of motion, in which the position of the robotic arm may be represented in joint space. In the context of this dissertation, considering the goals of reaching, grasping and transporting an object it is important to match the joint space with the cartesian space. Therefore, it is important to understand the theoretical definition of both direct and inverse kinematics.

Direct kinematics defines how to pass from joint space description to cartesian space by computing the position and orientation of the end-effector accordingly to the joints of the arm. On the contrary **Inverse Kinematics** is related to the opposite process, consisting in computing the value of all possible configurations for the joints of the arm, that allow reaching a given position and orientation knowing the end-effectors desired position and orientation. Note that given the non-linearity of the equations, calculating the inverse kinematics, may be impossible in some situations. These relations are expressed in figure 3.1:

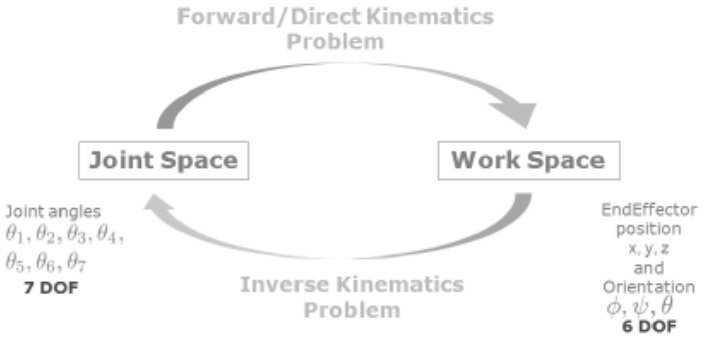


Figure 3.1: Figure expressing the relations between direct and inverse kinematics.

Keeping in mind that the robotic arms used can be described as a set of joints connected by links, their kinematics may be described accordingly to **Denavit-Hartenberg** parameters (a_i,α_i,d_i,θ_i):

- a_{i-1} - Related to the length between $\hat{z}_{i-1}$ and $\hat{z}_i$ measured in $\hat{x}_{i-1}$;
- α_{i-1} - Related to the angle between $\hat{z}_{i-1}$ and $\hat{z}_i$ measured in $\hat{x}_{i-1}$;
- d_i - Related to the length between $\hat{x}_{i-1}$ and $\hat{x}_i$ measured in $\hat{z}_i$;
- θ_i - Related to the angle between $\hat{x}_{i-1}$ and $\hat{x}_i$ measured in $\hat{z}_i$.

The convention used to represent the coordinated frames $(\hat{x}_i, \hat{y}_i, \hat{z}_i)$ for each link is:

$\hat{x}_i$, is pointing along **Denavit-Hartenberg's** parameter a_i, from one joint to the next joint;

The origin of each frame is located in the intersection between a_{i-1} and the joint axis;

The $\hat{z}$ axis of a local frame points along the axis of the joint;

$\hat{y}_i$ is discovered using the right-hand rule.

The parameters above will be used to describe the transformation matrix from joint $i-1$ to i. As stated before, the transformation matrix is described by equation 3.1

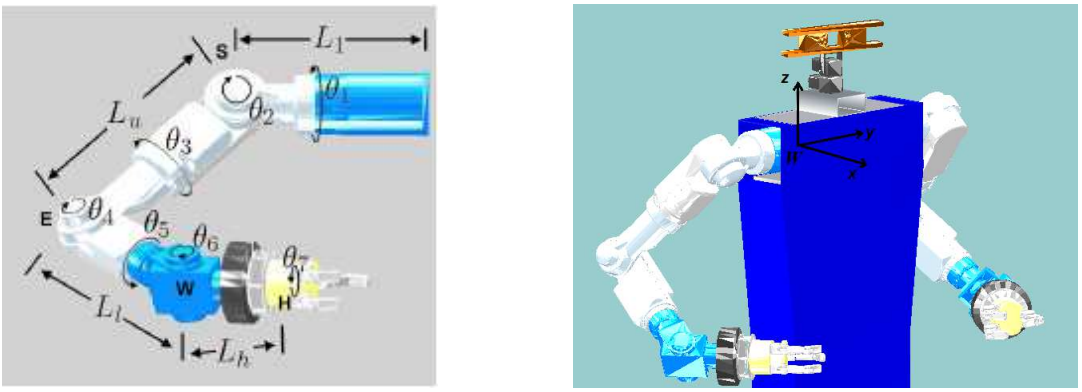
$$ {}^{i-1}_{i}T = Rot_X(\alpha_{i-1})\, Trans_X(a_{i-1})\, Trans_Z(d_i)\, Rot_Z(\theta_i) = \left[\begin{array}{c|c} Rot_{3\times3} & Trans_{3\times1} \\ \hline 0_{1\times3} & 1 \end{array}\right]. \tag{3.1} $$

Using this model of transformation matrix for each joint it is possible to compute direct kinematics for any arm of n joints as a sequence of multiplications given by equation 3.2:

$$ {}^{0}_{n}T = {}^{0}_{1}T\ {}^{1}_{2}T, \ldots, {}^{n-2}_{n-1}T\ {}^{n-1}_{n}T. \tag{3.2} $$

3.2 ARoS' Robotic Arms

Each **ARoS** robotic arm is a *Light-Weight Arm 7-DOF*, see fig. 3.2a). It has 7 revolute joints, weights 12 kg, it is able to lift a maximum payload of 3 kilograms and has a total length of 1200 mm. These characteristics are appropriated for the practical needs of this project, providing the means to perform all the movements required in the scenarios presented in section 7 and section 6.



(a) (b)

Figure 3.2: *Light Weight* arm mounted on **ARoS**. A representation of important points with joint rotations and arm lengths can be observed (Silva, 2011). The origin of the world with its axis located at the base of the right arm.

3.3 ARoS Right Arm

The robotic arm mounted on the right side of **ARoS** has its mathematical origin of the world located in its base (near the shoulder) as it is visible in the figure 3.2b. The main reason why the origin of the world is placed on the base of the right arm is reminiscent to the unimanual version of **ARoS**, it was designed this way mainly to ease the kinematics calculus for the right arm and in order to maintain the previously developed kinematics, the origin of the world was kept. Note that such decisions must be considered when implementing the kinematics for the left arm.

As seen on figure 3.2a the light-weighted arm has different rotation directions, arm lengths and different joint limits (as seen on table 3.1). One may see on table 3.2, that the robotic arm segments were stipulated with the main purpose to resemble the human arm, taking advantage of the fact that the robotic arm dimensions are proportional to the human arm. Accordingly, the following key points were established: **S** - Center of the Shoulder; **E** - Center of the Elbow; **W** - Center of the Wrist; **H** - Center of the palm of the hand.

Table 3.1: Joint limits of each joint of the right Light-Weight robotic arm.

Joint	Joint Limits (right arm)	Length (in mm)	Description
θ_1	[-165, 165]	200	Joint limits, distance from the origin to θ_1
θ_2	[-105, 91]	140	Joint limits, distance from θ_1 to θ_2
θ_3	[-165, 165]	200	Joint limits, distance from θ_2 to θ_3
θ_4	[-115, 106]	195	Joint limits, distance from θ_3 to θ_4
θ_5	[-165, 165]	205	Joint limits, distance from θ_4 to θ_5
θ_6	[-120, 120]	165	Joint limits, distance from θ_5 to θ_6
θ_7	[-165, 165]	95	Joint limits, distance from θ_6 to θ_7

Table 3.2: ARoS' robotic arm segments and relations to the human arm.

Arm Segment	Key Point	Length (in mm)	Description
L_1	**S**	340	Origin of the arm to the center of the shoulder
L_u	**E**	395	Length of the upper arm
L_l	**W**	370	Length of the lower arm
L_h	**H**	95	from the wrist until the end of the arm

3.3.1 Direct Kinematics of the right arm

To ease the calculus of the direct kinematics, the first step to consider is related to the position of the arm which is placed in a *zero position*, meaning that all joints have a zero value and an assigned coordinate frame (see fig: 3.3). These coordinate frames are assigned in conformity with the Denavit-Hartenberg convention.

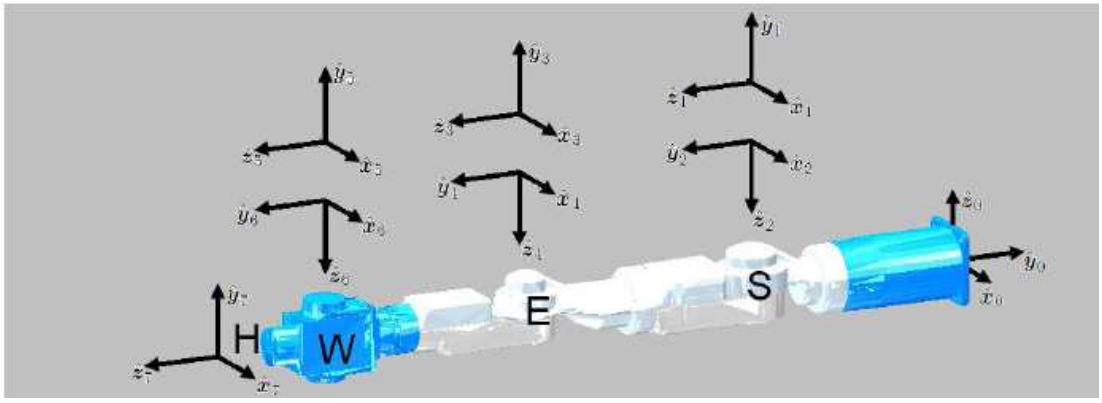


Figure 3.3: Left arm at zero position with the coordinate frames in which the Base is represented by $\hat{x}_0, \hat{y}_0, \hat{z}_0$, the Shoulder is represented by both θ_1 and θ_2 with their coordinate frames, the Elbow is represented by both θ_3 and θ_4 with their coordinate frames, the Wrist is represented by both θ_5 and θ_6 with theirs coordinate frames and the Hand is represented by θ_7 with its coordinate frames. Note: the origin of the world is in the base of the right arm (Silva, 2011).

The Denavit-Hartenberg parameters for each of the DOF's of the right arm is visible on table 3.3.

Table 3.3: Denavit-Hartenberg parameters for the right manipulator.

i	α_{i-1} (deg)	θ_i (deg)	d_i (mm)	a_{i-1} (mm)
1	90	θ_1	L_1	0
2	90	θ_2	0	0
3	-90	θ_3	L_u	0
4	90	θ_4	0	0
5	-90	θ_5	L_l	0
6	90	θ_6	0	0
7	-90	θ_7	L_h	0

Through the information taken from this table, it is possible to obtain all the transformation matrices for the direct kinematics of the right arm. The

transformation from the the base of the right arm (corresponding to the world origin W) to the first joint is seen on matrix 3.3:

$$ {}^{0}_{1}T = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) & 0 & 0 \\ 0 & 0 & -1 & -L_1 \\ \sin(\theta_1) & \cos(\theta_1) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.3} $$

The transformation for joint 2, 4, 6 is the following, where i=2,4,6 is seen on 3.4:

$$ {}^{i-1}_{i}T = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) & 0 & 0 \\ 0 & 0 & -1 & 0 \\ \sin(\theta_i) & \cos(\theta_i) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.4} $$

The transformation for joint 3, 5, 7 is the following, where i=3,5,7 and $L_3 = L_u$, $L_5 = L_l$ and $L_7 = L_h$ is visible on matrix 3.5

$$ {}^{i-1}_{i}T = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) & 0 & 0 \\ 0 & 0 & -1 & -L_i \\ -\sin(\theta_1) & -\cos(\theta_1) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.5} $$

The total transformations for the direct kinematics of the right arm, results in multiplying all the above presented transformation matrices. In the following equations, the origin of the world (W) refers to the base the arm (0). The transformations for the direct kinematics are visible in equation 3.6:

$$ {}^{W}_{7}T = {}^{W}_{1}T \; {}^{1}_{2}T \; {}^{2}_{3}T \; {}^{3}_{4}T \; {}^{4}_{5}T \; {}^{5}_{6}T \; {}^{6}_{7}T. \tag{3.6} $$

The relative posture of the arm segments can also be seen on equations 3.7, 3.8, 3.9 and 3.10:

$$ S = {}^{W}_{1}T \; S^1, \tag{3.7} $$

$$ E = {}^{W}_{3}T \; E^3, \tag{3.8} $$

$$W = {}^{W}_{5}T\ W^{5}, \tag{3.9}$$

$$H = {}^{W}_{7}T\ H^{7}. \tag{3.10}$$

In which $H^7 = W^5 = E^3 = S^1 = (0,0,0,1)^T$, once these are considered in the middle of the joints (see fig. 3.3). These equations are needed to obtain the direct kinematics equations (3.11, 3.12, 3.13 and 3.14) for x, y, z coordinates of each point.

$$S = (x_{shoulder}, y_{shoulder}, z_{shoulder})^T = \begin{bmatrix} 0 \\ -L_1 \\ 0 \end{bmatrix}, \tag{3.11}$$

$$E = (x_{elbow}, y_{elbow}, z_{elbow})^T = \begin{bmatrix} -L_u\ c_1\ s_2 \\ -L_1 - L_u\ c_2 \\ -L_u\ s_1\ s_2 \end{bmatrix}, \tag{3.12}$$

$$W = (x_{wrist}, y_{wrist}, z_{wrist})^T = \begin{bmatrix} L_l\ \xi_2 - L_u\ c_1\ s_2 \\ \psi_5\ L_l\ - L_u\ c_2 - L_1 \\ L_l\ \xi_4 - L_u\ s_1\ s_2 \end{bmatrix}, \tag{3.13}$$

$$H = (x_{hand}, y_{hand}, z_{hand})^T = \begin{bmatrix} (-(\xi_1\ c_5 - \psi_3\ s_5)\ s_6 + \xi_2\ c_6)\ L_h + \xi_2\ L_l - c_1\ s_2\ L_u \\ (-\xi_5\ s_6 + \psi_5\ c_6)\ L_h + \psi_5\ L_l - c_2\ L_u - L_1 \\ (-(\xi_3\ c_5 - \psi_4\ s_5)\ s_6 + \xi_4\ c_6)\ L_h + \xi_4\ L_l - s_1\ s_2\ L_u \end{bmatrix}. \tag{3.14}$$

The planning to be constantly aware of the location of each point of the arm, and verify if any of them collides with an obstacle. The location of the hand, also let know in which position the end-effector (palm of the hand) is and if it reaches the target object.

In order to obtain the hand orientation one must know the equation of the local frame $\hat{x}_7, \hat{y}_7, \hat{z}_7$, which can be obtained from the transformation matrix of

the Hand H, given by matrices 3.15, 3.16 and 3.17:

$$\hat{x}_7 = \begin{bmatrix} ((\xi_1\ c_5 - \psi_3\ s_5)\ c_6 + \xi_2\ s_6)\ c_7 - (\xi_1\ s_5 + \psi_3\ c_5)\ s_7 \\ (\xi_5\ c_6 + \psi_5\ s_6)c_7 - \xi_6\ s_7 \\ ((\xi_3\ c_5 - \psi_4\ s_5)\ c_6 + \xi_4\ s_6)\ c_7 - (\xi_3\ s_5 + \psi_4\ c_5)\ s_7 \end{bmatrix}, \tag{3.15}$$

$$\hat{y}_7 = \begin{bmatrix} -((\xi_1\ c_5 - \psi_3\ s_5)\ c_6 + \xi_2\ s_6)\ s_7 - (\xi_1\ s_5 + \psi_3\ c_5)\ c_7 \\ -(\xi_5\ c_6 + \psi_5\ s_6)s_7 - \xi_6\ c_7 \\ -((\xi_3\ c_5 - \psi_4\ s_5)\ c_6 + \xi_4\ s_6)\ s_7 - (\xi_3\ s_5 + \psi_4\ c_5)\ c_7 \end{bmatrix}, \tag{3.16}$$

$$\hat{z}_7 = \begin{bmatrix} -(\xi_1\ c_5 - \psi_3\ s_5)\ s_6 + \xi_2\ c_6 \\ -(\xi_5\ s_6 + \psi_5\ c_6) \\ -(\xi_3\ c_5 - \psi_4\ s_5)\ s_6 + \xi_4\ c_6 \end{bmatrix}. \tag{3.17}$$

In which the variables presented from equation 3.11 to equation 3.17, are defined as following:

$$\psi_1 = c_1\ c_2\ c_3 - s_1\ s_3 \qquad \xi_1 = \psi_1\ c_4 - c_1\ s_2\ s_4$$
$$\psi_2 = s_1\ c_2\ c_3 + c_1\ s_3 \qquad \xi_2 = -\psi_1\ s_4 - c_1\ s_2\ c_4$$
$$\psi_3 = c_1\ c_2\ s_3 + s_1\ c_3 \qquad \xi_3 = \psi_2\ c_4 - s_1\ s_2\ s_4$$
$$\psi_4 = s_1\ c_2\ s_3 - c_1\ c_3 \qquad \xi_4 = -\psi_2\ s_4 - s_1\ s_2\ c_4$$
$$\psi_5 = s_2\ c_3\ s_4 - c_2\ c_4 \qquad \xi_5 = \psi_6\ c_5 + s_2\ s_3\ s_5$$
$$\psi_6 = -s_2\ c_3\ c_4 - c_2\ s_4 \qquad \xi_6 = \psi_6\ s_5 - s_2\ s_3\ c_5$$

With the purpose to obtain the relation between the hand orientation and the object orientation, it is needed to relate both local frames, as seen on figure 3.4.

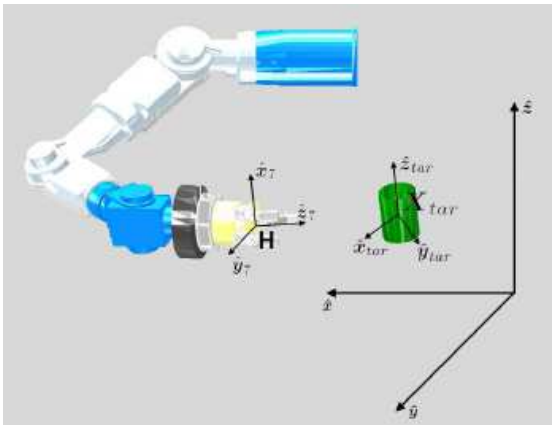


Figure 3.4: The local frames of the hand and target. These give information about the position of the hand, the target and also the relative orientation of one related to other (given by the $\hat{x}_7, \hat{y}_7, \hat{z}_7$ vectors and the $\hat{x}_{tar}, \hat{y}_{tar}, \hat{z}_{tar}$ vectors).

These relations vary accordingly to the grip type used, as will be explained in more detail at sections 6.3 and 7.2.

3.3.2 Inverse Kinematics of the right arm

If it is intended to compute the arm joint angles $\theta = (\theta_1, \ldots, \theta_7)^T$ of the arm that puts the hand (end-effector) at the desired position and orientation $\mathbf{X}_H = (x_H, y_H, z_H, \phi_H, \psi_H, \theta_H)^T$, it involves an inverse kinematics problem.

3.3.2.1 Solving the redundancy problem

The first aspect to consider is the redundancy of the **ARoS'** robotic arm caused by the free movement of the elbow when the hand is fixed (see fig. 3.5. This results on having a virtual infinite amount of possible solutions and can be solved by fixing a value for α (the angle between the plane that includes the shoulder, elbow and wrist, and a plane that contains the desired position of the elbow). This reduces the problem to one solution for each value of α.

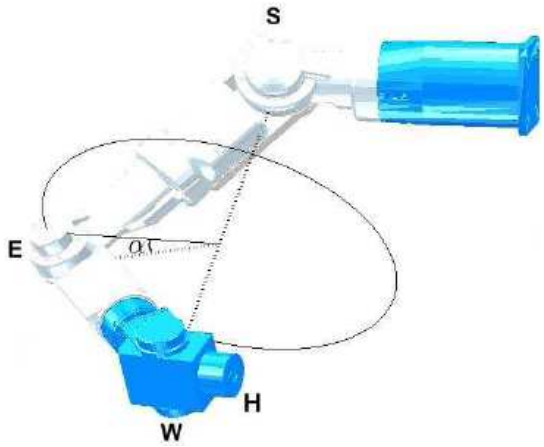


Figure 3.5: The **ARoS** arm allows for a redundancy corresponding to a free movement of the elbow, in which it freely rotate in a circular arc (Silva, 2011).

3.3.2.2 Obtaining the wrist position and orientation

The position and orientation of the target object is described by equation 3.18:

$$\mathbf{X}_{tar} = (x_{tar}, y_{tar}, z_{tar}, \phi_{tar}, \psi_{tar}, \theta_{tar})^T. \tag{3.18}$$

In order to respect the relation between the orientation of the hand and the orientation of the object to be grasped, the position of both palm of the hand and wrist are respectively described in equations 3.19 and 3.20 as the following:

$$\mathbf{H} = (x_{tar}, y_{tar}, z_{tar})^T + \hat{x}_{tar}\ L_{obj}. \tag{3.19}$$

and,

$$\mathbf{W} = H + \hat{x}_{tar}\ L_h. \tag{3.20}$$

where $\hat{x}_{tar}$ is obtained accordingly to equation 3.21,

$$\hat{x}_{tar} = (\cos(\phi_{tar}) \cdot \cos(\psi_{tar}), \sin(\phi_{tar}) \cdot \cos(\psi_{tar}), -\sin(\psi_{tar}))^T. \tag{3.21}$$

and L_{obj} is considered the distance between the palm of the hand and the object center.

3.3.2.3 Obtaining θ_4

This is the first joint to be computed since it is independent of redundancy caused by α. Regarding that θ_4 corresponds to the human elbow, it is logical to assume that it will only bend forward, as the human arm (see fig. 3.6). Therefore and applying the *Co-sines Law* the following solution of θ_4 is obtained accordingly to the equation 3.22, in which L_{SW} is the length between the shoulder and the wrist.

$$\theta_4 = -\arccos\left(\frac{-(L_u)^2 - (L_l)^2 + (L_{SW})^2}{2\ L_u\ L_l}\right). \tag{3.22}$$

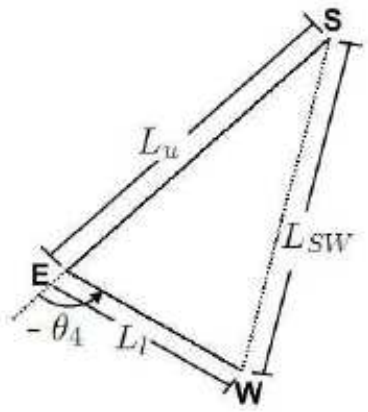


Figure 3.6: The cosine law can be used in order to obtain a solution for θ_4 by defining a triangle between the shoulder (S), the elbow (E) and the wrist (W).

3.3.2.4 Obtaining the position and orientation of the elbow from α

It is known that the elbow has a virtual infinite number of possible positions as a consequence from the variations of the value of α. Defining the $\hat{v}_{SW}$ as the unit vector from the shoulder to the wrist and $\hat{u}$ as the normal vector to $\hat{v}_{SW}$ parallel to the xWy plane (see fig. 3.7), the $\hat{v}$ vector is obtained accordingly to equation 3.23:

$$\hat{v} = \hat{u} \times \hat{v}_{SW}. \tag{3.23}$$

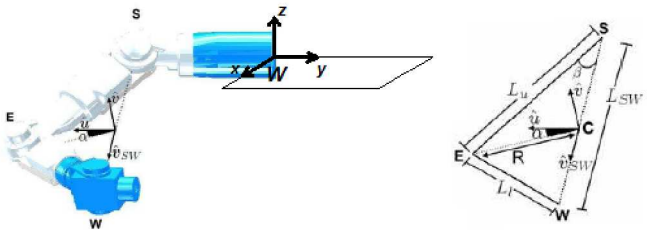


Figure 3.7: In this figure it is visible the center of redundancy with its local frame defined by the vectors ($\hat{\mathbf{u}}$,$\hat{\mathbf{v}}$,$\mathbf{v}_{\hat{SW}}$).

As displayed in figure 3.7 the center and the radius of the circumference are respectively obtained accordingly to the equations 3.24 and 3.25:

$$\mathbf{C} = \mathbf{S} + \cos(\beta)\ L_u\ \hat{v}_{SW}, \tag{3.24}$$

$$\mathbf{R} = L_u\ \sqrt{1 - (\cos^2(\beta))}. \tag{3.25}$$

In which $\cos(\beta)$ (visible in equation 3.26) is obtained using the cosine law, resulting in:

$$\cos(\beta) = \frac{(L_u)^2 - (L_l)^2 + (L_{SW})^2}{2\ L_u\ L_{SW}}. \tag{3.26}$$

With the above information it is possible to obtain the elbow position given by the equation 3.27:

$$\mathbf{E} = C + R\ (\cos(\alpha)\ \hat{u} + \sin(\alpha)\ \hat{v}). \tag{3.27}$$

Knowing the position of the elbow allows to obtain its orientation given by the vectors $\hat{\mathbf{x}}_4$, $\hat{\mathbf{y}}_4$, $\hat{\mathbf{z}}_4$ through the following equations:

$$\hat{\mathbf{z}}_4 = \hat{v}_{SW} \times \hat{v}_{CE}, \tag{3.28}$$

$$\hat{\mathbf{y}}_4 = \hat{v}_{EW}, \tag{3.29}$$

$$\hat{\mathbf{x}}_4 = \hat{y}_4 \times \hat{z}_4. \tag{3.30}$$

3.3.2.5 Obtaining the solutions for $\theta_1, \theta_2, \theta_3$

Having obtained the respective vectors, presented in equations 3.28, 3.29 and 3.30, the 3×3 rotation matrix from the base of the arm to joint 4 (θ_4) may be

defined as the following:

$$ {}^{0}_{4}\mathbf{R} = \left[\hat{x}_4|\hat{y}_4|\hat{z}_4.\right] \tag{3.31} $$

Furthermore, it is also possible to obtain the rotation matrix from the 3rd to the 4th joint (see equation 3.4), after having obtained θ_4 (equation 3.22). The rotation matrix is visible in matrix 3.32:

$$ {}^{0}_{4}\mathbf{R} = \begin{bmatrix} \cos(\theta_4) & -\sin(\theta_4) & 0 \\ 0 & 0 & -1 \\ \sin(\theta_4) & \cos(\theta_4) & 0 \end{bmatrix}. \tag{3.32} $$

Knowing that:

$$ {}^{0}_{4}\mathbf{R} = {}^{0}_{3}\mathbf{R}\ {}^{3}_{4}\mathbf{R}. \tag{3.33} $$

One may obtain the following solution for the rotation matrix from the base of the arm to the third joint:

$$ {}^{0}_{3}\mathbf{R} = {}^{0}_{4}\mathbf{R}\ ({}^{3}_{4}\mathbf{R})^{T}. \tag{3.34} $$

From equation 3.34, it is possible to extract two solutions for θ_2 as seen on equation 3.35:

$$ \theta_2 = \arctan_2\left(\pm\sqrt{1 - {r_{23}}^2}, -r_{23} \right). \tag{3.35} $$

For each of these solutions θ_1 and θ_3 are obtained as seen on equations 3.36 and 3.37:

$$ \theta_1 = \arctan_2\left(\frac{-r_{33}}{\sin(\theta 2)}, \frac{-r_{13}}{\sin(\theta 2)} \right), \tag{3.36} $$

$$ \theta_3 = \arctan_2\left(\frac{r_{22}}{\sin(\theta 2)}, \frac{-r_{21}}{\sin(\theta 2)} \right). \tag{3.37} $$

Taking into account what has been obtained above it becomes clear that if $\sin(\theta_2) = 0$, a singularity point is reached. To avoid the singularity problem, θ_1 is set to 0 and θ_3 defines the upper-arm orientation ($\theta_3 = \arctan_2(r_{31}, r_{32})$).

3.3.2.6 Obtaining the solutions for $\theta_5, \theta_6, \theta_7$

Knowing the end-effector respective vectors, presented in the equations 3.15, 3.16, 3.17, one may define the 3×3 rotation matrix from the base of the arm to joint

7 (θ_7) as the following:

$$^0_7\mathbf{R} = \left[\hat{x}_7|\hat{y}_7|\hat{z}_7.\right] \tag{3.38}$$

Knowing that:

$$^0_7\mathbf{R} = {}^0_4\mathbf{R}\ {}^4_7\mathbf{R} \tag{3.39}$$

Obtaining the following solution for the rotation matrix from the forth joint to the seventh joint as is shown here:

$$^4_7\mathbf{R} = ({}^0_4\mathbf{R})^T\ {}^0_7\mathbf{R} \tag{3.40}$$

From the equation 3.40, two solutions for θ_6 as seen on equation 3.41:

$$\theta_6 = \arctan_2\left(\pm\sqrt{1-{r_{23}}^2}, r_{23}\right). \tag{3.41}$$

For each of these solutions θ_5 and θ_7 are obtained as seen on equations 3.42 and 3.43:

$$\theta_5 = \arctan_2\left(\frac{r_{33}}{\sin(\theta_6)}, \frac{-r_{13}}{\sin(\theta_6)}\right), \tag{3.42}$$

$$\theta_7 = \arctan_2\left(\frac{-r_{22}}{\sin(\theta_6)}, \frac{r_{21}}{\sin(\theta_6)}\right). \tag{3.43}$$

Taking into account the obtained above it is clear that if $\sin(\theta_6) = 0$, a singularity point is reached. In this case, to avoid the singularity problem, θ_5 is set to 0 and θ_7 defines the hand orientation ($\theta_7 = \arctan_2(r_{12}, r_{32})$).

3.4 ARoS's Left Arm

It was previously affirmed that one important purpose of this dissertation was to endow the left arm with a successful movement planning. Accordingly, some changes in the mathematical model, had to be conducted in relation to the right arm. The new model for the left arm uses the same robotic manipulator as the right arm, which means that they share the same basic characteristics(e.g. world origin, dimensions, number of joints, among others, see figures 3.2a, 3.2b and table 3.2).

The first change concerns the orientation of the left arm. The main reason is that if the vertical orientation was kept (related to the right arm), this would result in the same seen movements of the right arm but in the opposite direction as can be seen on figure 3.8.

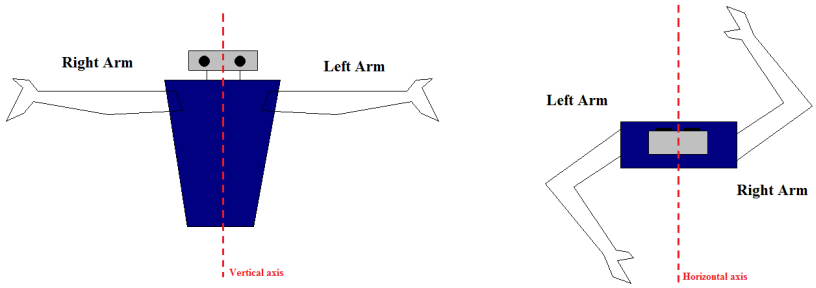


Figure 3.8: Non-inverted left arm. Visually pleasant (same posture seen on the right arm) but the moves would be done backwards which is unacceptable.

This means that the left arm would move symmetrically with respect to the origin, instead of being mirrored. The vertical orientation of the arm cannot be kept on the left side, as visible on figure 3.9. These corresponds to a rotation on the left arm along the x axis of the world.

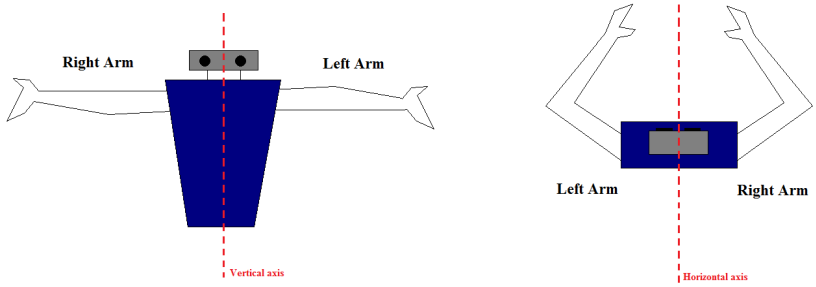


Figure 3.9: Inverting the orientation of the left arm. Visually less natural but moves the same way as the right arm (forward and without joint limit problems).

The previous explanation shows why it was important to change the orientation of the left arm when it was assembled. Due to the rotation done along the x axis of the world, the direction of rotation of all the joints is inverted (see table 3.4 in section 3.2). However to keep the mirrored behavior of the arm, it is required that the even joints (θ_2, θ_4, θ_6) maintain its joints range. On the other

hand it is important to mathematically consider the odd joint ranges (θ_1, θ_3, θ_5 and θ_7) to be inverted, otherwise those joints would reach their limits and be unable to accomplish the equivalent behavior of the right arm (see fig. 3.10).

Figure 3.10: Having applied a rotation of the left arm along the x axis it is obtained the showed configuration. The even joints (marked with orange color) maintain their range while the odd joints (marked with yellow color) need to have their range and limits inverted.

Table 3.4: Joint limits of each joint of the Light-Weight robotic arm.

Joints	Joints range (Right arm)	Joint range (Left arm)	Description
θ_1	[-165, 165]	[165, -165]	Range of limits of θ_1 for left and right arm
θ_2	[-115, 106]	[-115, 106]	Range of limits of θ_2 for left and right arm
θ_3	[-165, 165]	[165, -165]	Range of limits of θ_3 for left and right arm
θ_4	[-115, 106]	[-115, 106]	Range of limits of θ_4 for left and right arm
θ_5	[-165, 165]	[165, -165]	Range of limits of θ_5 for left and right arm
θ_6	[-120, 120]	[-120, 120]	Range of limits of θ_6 for left and right arm
θ_7	[-165, 165]	[165, -165]	Range of limits of θ_7 for left and right arm

As observable in table 3.4 the range of the odd joints has been mathematically inverted in order to keep the same behavior that it would have in the right side of **ARoS**. The practical consequence of this inversion is that any inserted joint value (for odd joints) is going to be inverted (e.g. if it is wanted that the joint 1 have a value of orientation of 65 degrees the real value sent is -65 degrees).

The second change concerns the imposed translation of the left arm in relation to the right arm. As stated in section 3.2 the origin of the world is located at the base of the right arm. This translation had to be conducted once the base of the left arm has not the same base position as the right arm, instead it is located 100 mm along the y axis (L_{W0}).

3.4.1 Direct Kinematics of the Left Arm

In the same way as the right arm direct kinematics, the arm is placed in a *zero position*, meaning that all joints have a zero value and an assigned coordinate frame, (see fig: 3.11). These coordinate frames are assigned in conformity with the Denavit-Hartenberg convention.

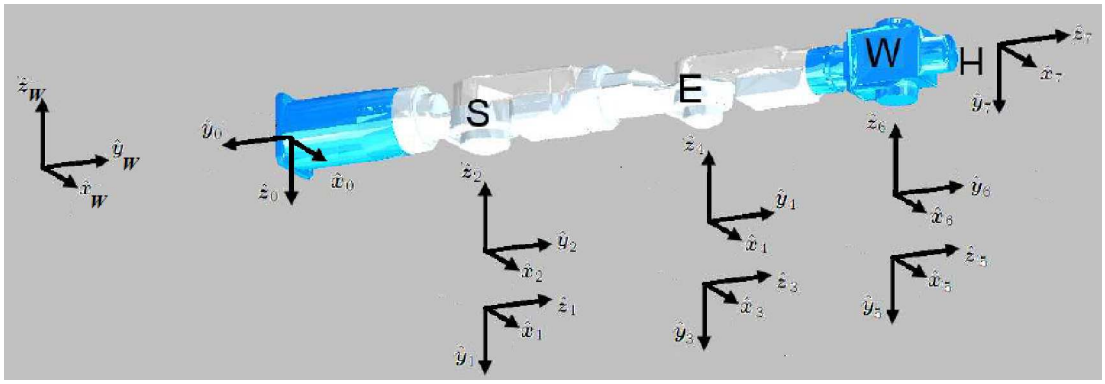


Figure 3.11: Left arm at zero position with the coordinate frames in which the Base is represented by $\hat{x}_0, \hat{y}_0, \hat{z}_0$, the Shoulder is represented by both θ_1 and θ_2 with their coordinate frames, the Elbow is represented by both θ_3 and θ_4 with theirs coordinate frames, the Wrist is represented by both θ_5 and θ_6 with theirs coordinate frames and the Hand is represented by θ_7 with its coordinate frames. Note: the origin of the world is represented by the coordinate frame $\hat{x}_W, \hat{y}_W, \hat{z}_W$.

It is noteworthy that the first row on table 3.5 does not refer to a joint but to the transformation required for using the left robotic arm, as mentioned in section 3.4.

Table 3.5: Denavit-Hartenberg parameters for the left manipulator.

i	α_{i-1} (deg)	θ_i (deg)	d_i (mm)	a_{i-1} (mm)
0	180	0	(L_{W0})	0
1	90	θ_1	L_1	0
2	90	θ_2	0	0
3	-90	θ_3	L_u	0
4	90	θ_4	0	0
5	-90	θ_5	L_l	0
6	90	θ_6	0	0
7	-90	θ_7	L_h	0

Through the information taken from table 3.5, it is possible to obtain all the transformation matrices for the direct kinematics of the left arm. The transformation from the world origin to the base of the left arm may be seen on matrix 3.44:

$$ {}^{W}_{0}T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(180) & \sin(180) & 100 \\ 0 & -\sin(180) & \cos(180) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.44} $$

The transformation from the the base of the left arm to the first joint is seen on matrix 3.45:

$$ {}^{0}_{1}T = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) & 0 & 0 \\ 0 & 0 & -1 & -L_1 \\ \sin(\theta_1) & \cos(\theta_1) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.45} $$

The transformation for joint 2, 4, 6 is the following, where i=2,4,6 is seen on 3.46:

$$ {}^{i-1}_{i}T = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) & 0 & 0 \\ 0 & 0 & -1 & 0 \\ \sin(\theta_i) & \cos(\theta_i) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.46} $$

The transformation for joint 3, 5, 7 is the following, where i=3,5,7 and $L_3 = L_u$, $L_5 = L_l$ and $L_7 = L_h$ is visible on matrix 3.47

$$ {}^{i-1}_{i}T = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) & 0 & 0 \\ 0 & 0 & -1 & -L_i \\ -\sin(\theta_1) & -\cos(\theta_1) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.47} $$

The total transformations for the direct kinematics of the left arm, results in multiplying all the above presented transformation matrices. The transformations for the direct kinematics are visible in equation 3.48:

$$ {}^{W}_{7}T = {}^{W}_{0}T\ {}^{0}_{1}T\ {}^{1}_{2}T\ {}^{2}_{3}T\ {}^{3}_{4}T\ {}^{4}_{5}T\ {}^{5}_{6}T\ {}^{6}_{7}T. \tag{3.48} $$

The relative posture of the arm segments can also be seen on equations 3.49, 3.50, 3.51 and 3.52:

$$ S = {}^{W}_{1}T\ S^1, \tag{3.49} $$

$$ E = {}^{W}_{3}T\ E^3, \tag{3.50} $$

$$ W = {}^{W}_{5}T\ W^5, \tag{3.51} $$

$$ H = {}^{W}_{7}T\ H^7. \tag{3.52} $$

In which $H^7 = W^5 = E^3 = S^1 = (0, 0, 0, 1)^T$, once these are considered in the middle of the joints (see fig. 3.11). These equations are important in order

to obtain the direct kinematics equations (3.53, 3.54, 3.55 and 3.56) for x, y, z coordinates of each point.

$$S = (x_{shoulder}, y_{shoulder}, z_{shoulder})^T = \begin{bmatrix} 0 \\ L_1 + 100 \\ 0 \end{bmatrix}, \tag{3.53}$$

$$E = (x_{elbow}, y_{elbow}, z_{elbow})^T = \begin{bmatrix} -L_u\ c_1\ s_2 \\ L_1 + L_u\ c_2 + 100 \\ L_u\ s_1\ s_2 \end{bmatrix}, \tag{3.54}$$

$$W = (x_{wrist}, y_{wrist}, z_{wrist})^T = \begin{bmatrix} L_l\ \mu_1 - L_u\ c_1\ s_2 \\ L_1 + L_l\ \mu_2 + L_u\ c_2 + 100 \\ L_l\ \mu_3 + L_u\ s_1\ s_2 \end{bmatrix}, \tag{3.55}$$

$$H = (x_{hand}, y_{hand}, z_{hand})^T = \begin{bmatrix} L_h\ \mu_4 + L_1\ \mu_5 - L_u\ c_1\ s_2 \\ L_1 + L_l\ \mu_2 - L_h\ \mu_6\ L_u\ c_2 + 100 \\ L_h\ \mu_7 + L_l\ \mu_3 + L_u\ s_1\ s_2 \end{bmatrix}. \tag{3.56}$$

Similarly to what happens with the right arm kinematics, the above coordinates allows the planning to be constantly aware of the location of each point of the arm, and verify if any of them collides with an obstacle. The location of the hand, also let know in which position the end-effector (palm of the hand) is and if it reaches the target object.

In order to obtain the hand orientation one must know the equation of the local frame $\hat{x}_7, \hat{y}_7, \hat{z}_7$, which can be obtained from the transformation matrix of the Hand H, given by matrices 3.57, 3.58 and 3.59:

$$\hat{x}_7 = \begin{bmatrix} c_7\ \mu_9 + s_7\ \mu_1 0 \\ c_7\ o_{11} - s_7\ o_{12} \\ c_7\ o_{13} + s_7\ o_{14} \end{bmatrix}, \tag{3.57}$$

$$\hat{y}_7 = \begin{bmatrix} c_7\ o_{15} - s_7\ o \\ -c_7\ o_{12} - s_7\ o_{11} \\ c_7\ o_3 - s_7\ o_4 \end{bmatrix}, \tag{3.58}$$

$$\hat{z}_7 = \begin{bmatrix} c_6\ \mu_5 + s_6\ o_6 \\ c_6\ \mu_2 - s_6\ o_8 \\ c_6\ \mu_3 + s_6\ o_{10} \end{bmatrix}. \tag{3.59}$$

In which the variables presented from equation 3.53 to equation 3.59, are defined as following:

$\mu_1 = s_4\ s_1\ s_3 - c_1\ c_2\ c_3 - c_1\ c_4\ s_2$

$\mu_2 = c_2\ c_4 - c_3\ s_2\ s_4$

$\mu_3 = s_4\ (c_1\ s_3 + c_2\ c_3\ s_1) + c_4\ s_1\ s_2$

$\mu_4 = c_6\ \mu_5 + s_6\ o_6$

$\mu_5 = s_4\ (s_1\ s_3 - c_1\ c_2\ c_3) - c_1\ c_4\ s_2$

$\mu_6 = s_6\ o_8 - c_6\ \mu_2$

$\mu_7 = c_6\ \mu_3 + s_6\ o_{10}$

$\mu_8 = c_4\ (s_1\ s_3 - c_1\ c_2\ c_3) + c_1\ s_2\ s_4$

$\mu_9 = s_6\ \mu_5 - c_6\ o_6$

$\mu_{10} = s_5\ \mu_8 - c_5\ (c_3\ s_1 + c_1\ c_2\ s_3)$

$o_{11} = c_6\ o_8 + s_6\ \mu_2$

$o_{12} = s_5\ (c_2\ s_4 + c_3\ c_4\ s_2) + c_5\ s_2\ s_3$

$o_{13} = s_6\ \mu_3 - c_6\ o_{10}$

$o_{14} = s_5\ o_{16} - c_5\ (c_1\ c_3 - c_2\ s_1\ s_3)$

$o_{15} = s_5\ \mu_8 - c_5\ (c_3\ s_1 + c_1\ c_2\ s_3)$

$o = s_6\ \mu_5 - c_6\ o_6$

$o_3 = s_5\ o_{16} - c_5\ (c_1\ c_3 - c_2\ s_1\ s_3)$

$o_4 = s_6\ \mu_3 - c_6\ o_{10}$

$o_6 = c_5\ \mu_8 + s_5\ (c_3\ s_1 + c_1\ c_2\ s_3)$

$o_8 = c_5\ (c_2\ s_4 + c_3\ c_4\ s_2) - s_2\ s_3\ s_5$

$o_{10} = c_5\ o_{16} + s_5\ (c_1\ c_3 - c_2\ s_1\ s_3)$

$o_{16} = c_4\ (c_1\ s_3 + c_2\ c_3\ s_1) - s_1\ s_2\ s_4$

Once again, it is needed to relate both local frames, as seen on figure 3.12, with the purpose to obtain the relation between the hand orientation and the object orientation.

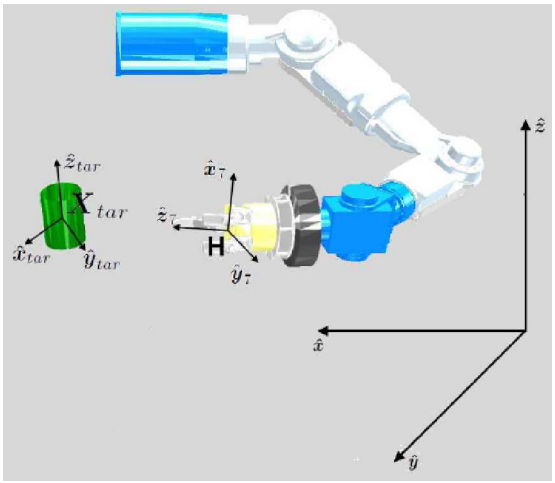


Figure 3.12: The local frames of the hand and target. These give information about the position of the hand, the target and also the relative orientation of one related to other (given by the $\hat{x}_7, \hat{y}_7, \hat{z}_7$ vectors and the $\hat{x}_{tar}, \hat{y}_{tar}, \hat{z}_{tar}$ vectors).

These relations vary accordingly to the grip type used, as will be explained in more detail at section 6.3 and section 7.2.

3.4.2 Inverse Kinematics of the left arm

Asserting the arm joint angles $\theta = (\theta_1, \ldots, \theta_7)^T$ for the desired position and orientation (see equation 3.60) of the hand (end-effector), involves solving an inverse kinematics problem.

$$\mathbf{X}_H = (x_H, y_H, z_H, \phi_H, \psi_H, \theta_H)^T. \tag{3.60}$$

3.4.2.1 From Right to Left arm Inverse Kinematics

The kinematics of the left arm uses the same models as the right arm kinematics, nevertheless, some changes must be conducted.

- Due to the transformation of the left arm explained in section 3.4, an equivalent translation has to be performed for the target objects. The rotation along the x-axis corresponds to invert the y value of the target location as may be seen on figure 3.13. The translation along the y-axis corresponds of

adding the distance between the world origin and the base of the arm (L_{W0}) to the y-value of the target location. Therefore the position and orientation of the target is given by:

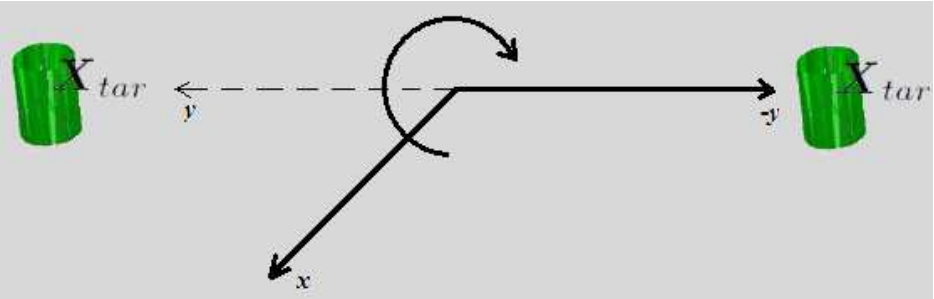


Figure 3.13: Inversion of the y value due to the rotation along the x axis.

$$\mathbf{X}_{tar} = (x_{tar}, \boldsymbol{-y_{tar} + L_{W0}}, z_{tar}, \phi_{tar}, \psi_{tar}, \theta_{tar})^T. \tag{3.61}$$

- After calculating the joint values using the right arm inverse kinematics for the new target, the odd joints must be inverted as refered in section 3.4, table 3.4. This means that $\theta_1 = -\theta_1$, $\theta_3 = -\theta_3$, $\theta_5 = -\theta_5$ and $\theta_7 = -\theta_7$ in order to be correctly adapted for the left arm.

3.5 ARoS' Robotic Hands

As mentioned in section 1.1.7, the robotic dexterous hand used is a *BH-8-series* from Barret-Hand (see fig. 3.14). This robotic hand has three fingers, each one having 1 controlled DOF and there is also an additional DOF to spread finger 1 and finger 2.

3.5.1 Direct Kinematics of the Hands

The direct kinematics of the hand will be used to obtain points of the fingers (phalanges and tip of each finger) with the purpose to check for collisions with obstacles or contact with the target.

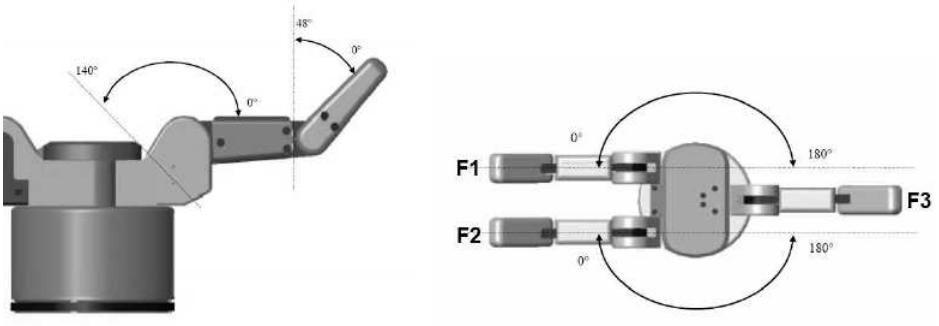


Figure 3.14: Here the *BH8-series Barret-Hand* used in both arms is observable. In the right arm, fingers F1 and F2 rotate synchronously around the palm of the hand with a limit of 180 degrees. Each finger has two joints limited to 140 and 48 degrees (Silva, 2011).

Using the Denavit-Hartenberg convention it is possible to define the direct kinematics of the robotic hand. As may be seen in figure 3.15 the joints of the hand can be referred as θ_8 (defining the spread) and $\theta_9, \theta_{10}, \theta_{11}$ (defining the DOF's of the three fingers k)

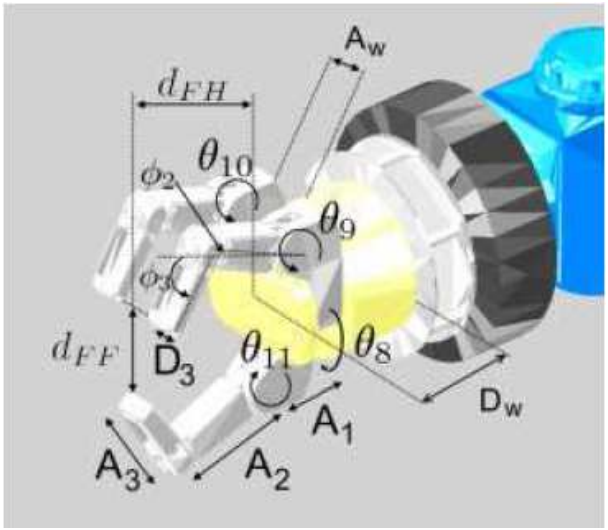


Figure 3.15: Posture of the BarretHand represented by $\theta_{hand} = (\theta_8, \theta_9, \theta_{10}, \theta_{11},)^T$ (Silva, 2011).

From this point it is possible to calculate the transformation matrix of the hand, as seen bellow:

$$ {}^{7}_{k,8}T = \begin{bmatrix} c_a & -s_a & 0 & r_k\ A_w \\ s_a & c_a & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad {}^{k,8}_{k,9}T = \begin{bmatrix} c_b & -s_b & 0 & A_1 \\ 0 & 0 & -1 & 0 \\ s_b & c_b & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, $$

$$ {}^{k,9}_{k,10}T = \begin{bmatrix} c_c & -s_c & 0 & A_2 \\ s_c & c_c & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad {}^{k,10}_{k,11}T = \begin{bmatrix} 1 & 0 & 0 & A_3 \\ 0 & 0 & 1 & D_3 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. $$

In which $a = r_k\theta_8 - 90\ j_k$, $b = \phi_2 + \theta_{8+k}$, $c = \phi_3 + 1/3\theta_{8+k}$, $k = (1, 2, 3)$, $r = (-1, 1, 0)^T$ and $j = (-1, -1, 1)^T$. With these matrices one can obtain the position of the finger tip k for each finger which is given by the equation 3.62:

$$ \mathbf{F}_k = \left({}^{0}_{7}T\ {}^{7}_{k,8}T\ {}^{k,8}_{k,9}T\ {}^{k,9}_{k,10}T\ {}^{k,10}_{k,11}T \right) {}^{11}_{k}\mathbf{F}. \tag{3.62} $$

In which ${}^{11}_{k}\mathbf{F} = (0, 0, 0, 1)^T$ because the finger tip position for each finger is on the same position as the last local frame (tip of the finger), accordingly there is no additional translation. Note that ${}^{0}_{7}T$ refers to the transformation of the arms described in section 3.4.1.

3.6 Applying the kinematics to the Control Interface

The previously developed interface (see fig. 3.16) allowed for manually control the right arm of **ARoS** and observe the unimanual movements performed during the simulations. Once this dissertation focus on endowing bimanual capabilities to **ARoS**, it became necessary to modify the control interface (both **ARoS** CAD model and the control panel) as visible in figure 3.17.

3.6.1 Control Interface for Unimanual Movements

The unimanual control interface was developed in Matlab® and is composed by a graphical window that has a CAD model of the unimanual **ARoS** and its workspace (see fig. 3.16). It uses graphical functions to simulate the movement of the arm that uses the direct and inverse kinematics for the right arm discussed in this chapter. The control interface is also composed by a control panel that allows for manual control of the arm (through a set of eleven sliders, one for each joint). Debug of both direct and inverse kinematics is possible by returning both visual (through the graphical window), and mathematical feedback (giving the position of the Elbow and both position and orientation of the Hand), visible in the pink rectangles. The redundancy of the inverse kinematics can also be tested through changes in the value of the *Alpha Slider* (visible bellow the pink boxes).

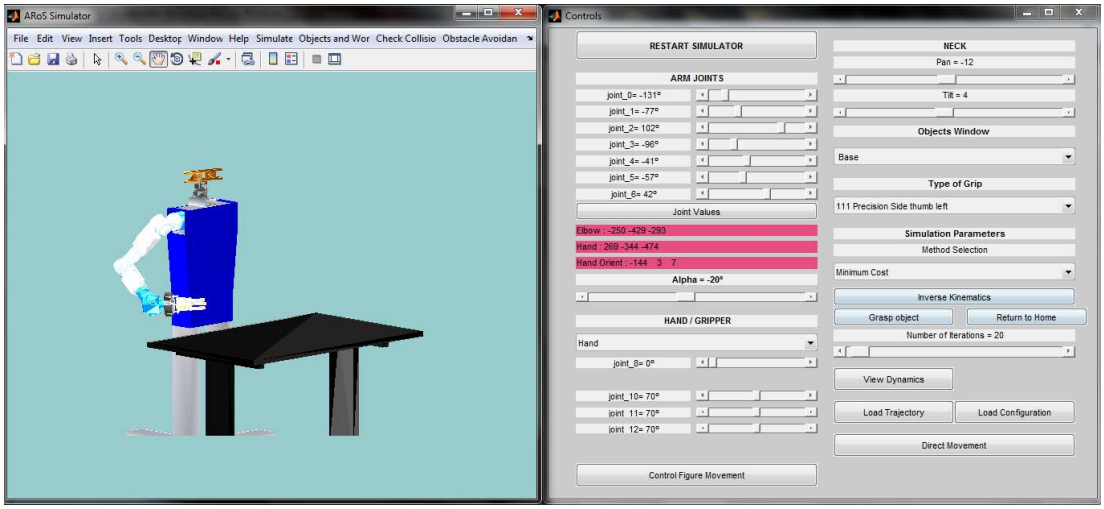


Figure 3.16: On the left side of the figure is visible the unimanual **ARoS**, modeled in CAD, in its workspace. On the right side of the figure is visible the control panel of the **ARoS** directed to unimanual movements, that uses the kinematics presented in this section.

3.6.2 Control Interface for Unimanual and Bimanual Movements

With the intent to have a bimanual control interface some changes were conducted. The main changes are the following:

- The model of the right arm was replicated applying the transformation discussed in section 3.4, the direct and inverse kinematics of the left arm, in order to get both mathematical and graphical coherency;
- For manually control and debug the left arm a new control area had to be developed in the control panel. This new area was named "Left Arm" and has the same debugging functions as the previously developed "Right Arm" area;
- In addition new mathematical feedback was implemented in order to better compare the position and orientation of both left and right arm's sections (Shoulder, Elbow, Wrist positions and Hand position and orientation).

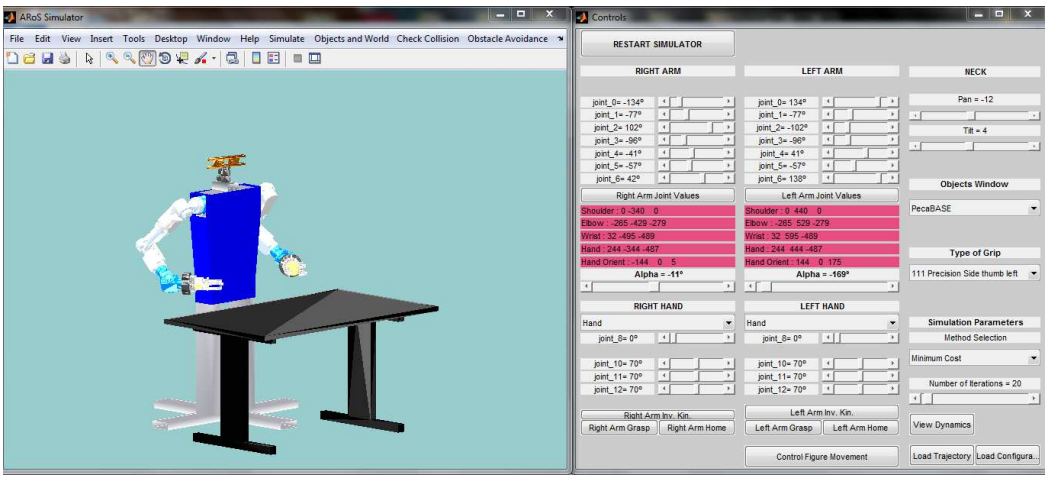


Figure 3.17: On the left side of the figure is visible the bimanual **ARoS**, modeled in CAD, in its workspace. On the right side of the figure is visible the control panel of the **ARoS** directed to bimanual movements, that uses the kinematics presented in this section.

Chapter 4

Non-linear Optimization applied to ARoS' Unimanual and Bimanual Movements

4.1 *Posture-based motion planning* for unimanual manipulation

This section describes the formalization of the object manipulation for unimanual robotic arm and hand movements inspired by the Posture-based motion planning (PBMP) model from (Rosenbaum et al., 2001).

The method developed for this work has, the basic principle of selecting a final and a bounce posture, in order to avoid obstacles before reaching the target (see fig: 4.1). Based on the ideas proposed by Rosenbaum, several assumptions are conducted for this method:

- **Goal Postures** are specified before the start of the movement. There are several ways in which the manipulator is able to reach the target (whether for grasping or manipulating an object), nevertheless only one is going to be used accordingly a set of constraints. These constraints depend of the

task performed (for example, not colliding the manipulator with the table or the robot body, grasping the target object perpendicularly to the hand, among others);

- The **Direct Kinematics** process (described in section 3.4.1) is used to determine the points of the arm and hand. These points refer to main points of the arm (shoulder, elbow, wrist and hand) and also intermediate points (point between shoulder and elbow, elbow and wrist, wrist and hand). The points of the fingers are also considered, but only during the grasping movements, the same happens to the object itself, which is considered a point during transport movements;

- The **Velocity** of the movement follows a bell-shaped profile, meaning that the movement starts and ends with zero velocity and acceleration, reaching its peak in the middle of the movement. Therefore allowing a natural and smooth movement.

One important process implemented is inspired in (Rosenbaum et al., 2001) and is related to the choice of goal posture candidates. For each specific task, there is a set of constraints (that take into account some features such as the height of the table, or position of the obstacles). The candidate postures must respect all these constraints and if one of them is violated the candidate is weeded out. The candidate that minimize the most the objective function while respecting the set of constraints becomes the final posture. This elimination process assures that a viable posture is chosen if there is a solution.

An important process described in (Rosenbaum et al., 2001) is the obstacle avoidance process, in which the planning is done in the joint space. In order to solve this problem, a solution was implemented that is related to the allowance of two simultaneous movements being performed by the manipulator: **1)** one is the main movement that corresponds to a direct movement from the starting position to the goal position with a bell-shaped velocity profile; **2)** the other movement is

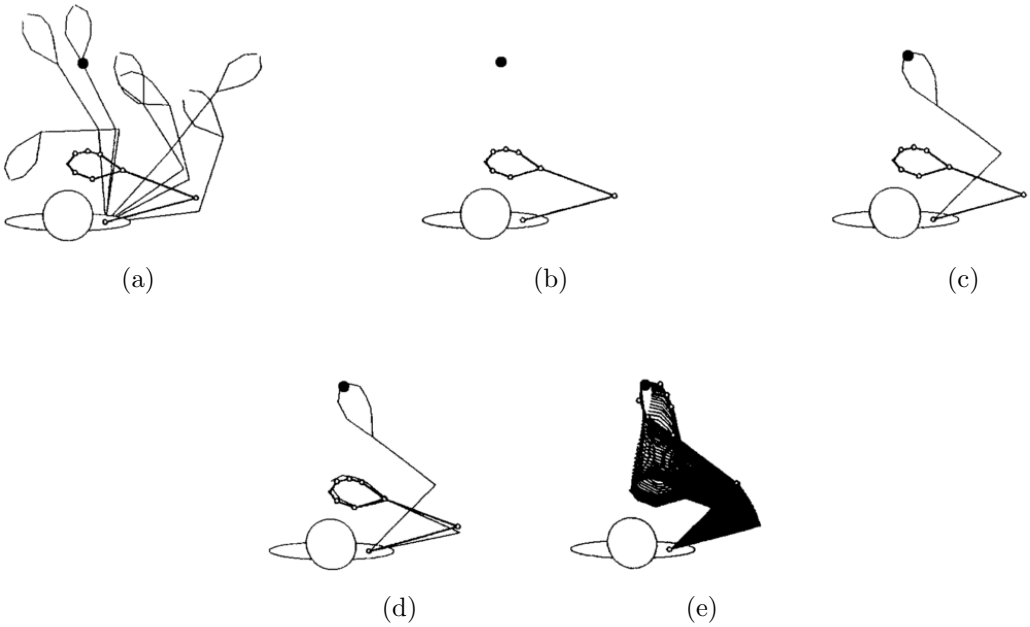


Figure 4.1: Visualization of a bounce and a final posture to avoid obstacles (Rosenbaum et al., 2001).

a detour from a starting position to a selected bounce posture and back to the starting position in order to contour the obstacles (back-and-forth movement). This allows the association between obstacles in the workspace and areas to avoid during the movement planning.

These two movements are superimposed in order to form a single movement with obstacle avoidance and target directed trajectory. The relevance of this method is that the back-and-forth movement (and consequently the selection of the bounce postures) is even more important than finding a direct movement.

The method described by Rosenbaum et al. (2001) is modeled for unimanual movements in human. In order to implement our solution for bimanual manipulation it is necessary to model the left arm movement, with the transformations explained in section 3.4. For this work, the movement of each arm is calculated separately. This helps understand why the resulting bimanual movements are asynchronous.

4.2 Description of the mathematical model for non-linear optimization

Accordingly to section 4.1, there are two sub-problems:

A Final posture problem (**Pa**), related to the final posture selection with the intent of obtaining the vector $\boldsymbol{\theta}_f \in \mathbb{R}^{n_j}$.

A Bounce posture problem (**Pb**), related to the bounce posture with the intent of obtaining the vector $\boldsymbol{\theta}_b \in \mathbb{R}^{n_j}$, where n_j is the number of joints.

There are several solutions described for each problem. For instance in Silva (2011) it was solved the bounce posture problem through an inverse kinematics approach. Following this approach, the solutions for this problem were tried iteratively until a sample of acceptable solutions were obtained which concluded the search. From the acquired sample, the solution with the minimum cost for all joints is selected and applied to the robotic arm movements. This process revealed two serious disadvantages:

- Once only a limited sample of possible solutions is obtained, there is no guarantee that this sample contains the optimal solution;
- Because this search is based on a trial-and-error iterative process and knowing that the inverse kinematics is a heavy mathematical problem, this process revealed to be a very time-consuming method (disallowing real-time Human-robot interaction).

Considering the disadvantages presented, in this dissertation these problems are formalized as two nonlinear optimization problems with simple bounds and equality and inequality constraints. This approach starts by defining the sequence of the eleven joint angles $(\theta_1, \ldots, \theta_{11})$ of the robotic arm and hand accordingly to the following:

$$\begin{aligned}\boldsymbol{\theta}(t, \boldsymbol{\theta}_f, \boldsymbol{\theta}_b) &= \boldsymbol{\theta}_0 + (\boldsymbol{\theta}_f - \boldsymbol{\theta}_0)\left(10\tau^3 - 15\tau^4 + 6\tau^5\right) + \boldsymbol{v}_0\, T\left(\tau - 6\tau^3 + 8\tau^4 - 3\tau^5\right) \\ &\quad + \frac{1}{2}\boldsymbol{a}_0\, T^2\left(\tau^2 - 3\tau^3 + 3\tau^4 - \tau^5\right) + (\boldsymbol{\theta}_b - \boldsymbol{\theta}_0)\sin^2(\pi\,\tau^{\vartheta}), \qquad (4.1)\end{aligned}$$

where $\boldsymbol{\theta}_0, \boldsymbol{v}_0, \boldsymbol{a}_0 \in \mathbb{R}^{n_j}$ represent the initial joint position, velocity and acceleration, T represents the movement duration, t is the movement time, $\tau = \frac{t}{T} \in [0, 1]$ is the normalized movement duration, and $\vartheta = -\frac{\ln 2}{\ln t_b}$, t_b is the movement time when the bounce posture is applied. The movement time $t \in [0, T]$ is discretized by N_T equally spaced points $t_i = i\ \delta$, where $\delta = \frac{T}{N_T}$ is the step size and $i = 0, 1, \ldots, N_T$. It was conventioned that $\boldsymbol{\theta}(t_i, \boldsymbol{\theta}_f, \boldsymbol{\theta}_b)$ represents $\boldsymbol{\theta}$ in equation 4.1 at time t_i.

The problems of selecting the final and bounce posture share some specifications:

- Firstly eleven joints are considered, in which seven are related to the arm $(\theta_1, \ldots, \theta_7)$ (see section 3.4.1) and four are associated to the hand $(\theta_8, \ldots, \theta_{11})$ (see section 3.4.1);

- For each robotic hand the action of grasping is only considered successful if finger 1 and finger 2 share the same joint values $\theta_{f,9} = \theta_{f,10}$ because we assume that they work synchronously, while the finger 3 (thumb) is independent to the other two fingers in some grip types. Therefore resulting in the calculus of just two fingers. Accordingly the total number of joints calculated (arm and hand) is 10 instead of 11, meaning that at most the hand has 3 joints in order to boost the calculus speed by reducing the dimension of the problem and without harming the grasping movement;

- It is worthwhile noting that in almost every movements the position of the thumb is the opposite of the two other fingers, that means that the spread value (θ_8) is zero. However this is not always verified because there is a specific grip in which the spread has a value different from zero (perpendicular to the other two fingers of 90 degrees). This happens for the *Side Grip Thumb-up* (see table 6.1).

4.2.1 Defining Arm/Hand points, Obstacles, Targets and body

This section describes how the movement planning interprets the scenario. These concepts are of outmost importance to integrate constraints and generated trajectories that are in accordance to the task and the scenario. The key elements are the following:

Arm/Hand points : The movement planning sees both arms and hands as a series of points modeled as spheres with variable diameters (see fig. 4.3b). These points are obtained through the calculus of the direct kinematics and are described in table 4.1.

Table 4.1: Arm/Hand points defined in both right and left arms.

i	Point	Description
1	$\frac{S+E}{2}$	Midpoint between Shoulder and Elbow
2	E	Elbow
3	$\frac{W+E}{2}$	Midpoint between Wrist and Elbow
4	W	Wrist
5	$W + 0.45 \times (H - W)$	Closer to Wrist, between Wrist and Hand
6	$W + 0.75 \times (H - W)$	Closer to Hand, between Wrist and Hand
7	F_{1_1}	First Phalanx of the finger 1
8	F_{2_1}	First Phalanx of the finger 2
9	F_{3_1}	First Phalanx of the finger 3
10	F_{1_2}	Second Phalanx of the finger 1
11	F_{2_2}	Second Phalanx of the finger 2
12	F_{3_2}	Second Phalanx of the finger 3
13	$F_{1_{tip}}$	Tip of the finger 1
14	$F_{2_{tip}}$	Tip of the finger 2
15	$F_{3_{tip}}$	Tip of the finger 3

Note that S, E, W, H contains the x, y, z coordinates of the center of the shoulder, elbow, wrist and hand respectively with the defined radius of the spheres that contain those points in the arm. In its turn, $F_{n,1}, F_{n,2}, F_{n,tip}$ define the x, y, z coordinates of the sections of each finger n (phalanges 1, 2 and tip), as it is visible in figure 4.2;

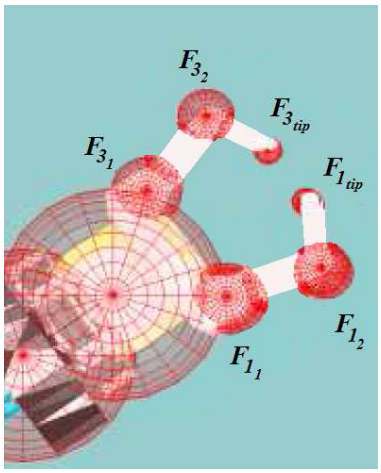
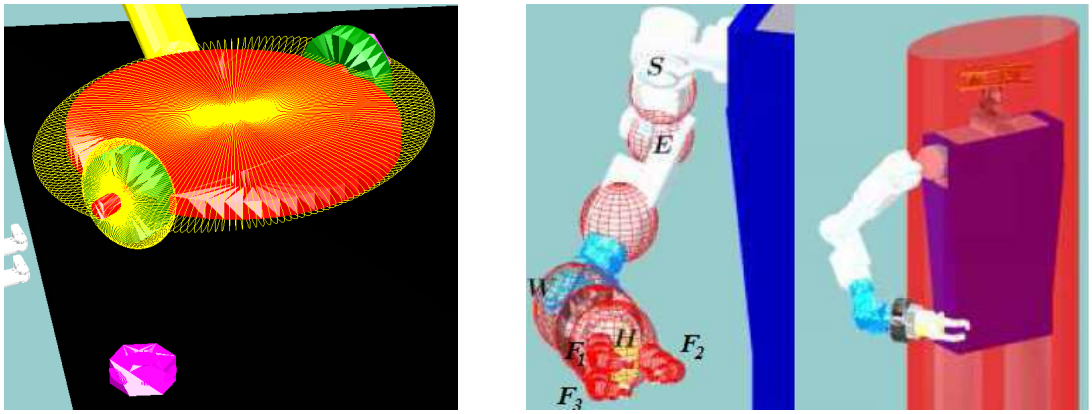


Figure 4.2: Points of each phalanges and tips in each finger of the BarretHand. As it is visible in this figure each point is modeled has a sphere.

Obstacles and Targets: Both obstacles and targets are modeled as ellipsoids with the dimensions in x, y, z obtained through the vision system (see fig. 4.3a). The planned movement must avoid contact of the obstacle ellipsoids with the points of the arm and hand, and should be directed to the target ellipsoid;

Body: Preventing collisions of the arms against the body is also important. For the movement planning the body is an elliptic cylinder that involves **ARoS'** (see fig. 4.3b). Both **ARoS** manipulators must avoid and stay outside of the elliptic cylinder in order to prevent collisions against the body.



(a) Ellipsoids containing objects of the scenario.

(b) Spheres containing body and arm points.

Figure 4.3: Ellipsoids containing objects and spheres containing arms and body. Note that in panel **a)** the obstacles are portrayed with a yellow ellipsoid and the target magenta's one. In panel **b)** the red spheres define the points for both arms and the body is defined with a red elliptic cylinder.

4.2.2 Final posture problem - (Pa)

In order to select the final posture a minimization problem must be solved, which is defined by non-linear functions, consisting of an objective function and a set of constraints on its variables. The optimization process tries to minimize the value of the objective function subject to all the defined constraints. Formulating the problem of selection of the final posture has a problem of non-linear optimization it is expected that the solution obtained is the optimal solution, because of the process of minimization conducted over its constraints.

One must know that all the calculus computed during the final posture problem tries to find a solution just for the seven joints of the arm $(\theta_1, \ldots, \theta_7)$ because the four joints of the hand $(\theta_8, \ldots, \theta_{11})$ are already predefined depending on the grip type of the movement and the size of the object to be grasped. However all the joints are considered for checking the constraints of the model that involve arm/hand points.

Accordingly, the final posture optimization problem (**Pa**) has the following objective function and set of constraints, in which $\theta_{f,k}$, where $k = 1, \ldots, 7$:

$$\mathbf{Pa} \min_{\boldsymbol{\theta}_f \in \mathbb{R}^7} \quad \sum_{k=1}^{7} \lambda_k \left(\theta_{0,k} - \theta_{f,k}\right)^2, \lambda_k \geq 0 \tag{4.2}$$

$$\text{s.t.} \quad \boldsymbol{c}_1(\boldsymbol{\theta}_f) = 0 \tag{4.3}$$

$$\|\boldsymbol{c}_2(\boldsymbol{\theta}_f)\|^2 \leq \delta \tag{4.4}$$

$$\boldsymbol{h}_f(\boldsymbol{\theta}_f) \leq \mathbf{0} \tag{4.5}$$

$$\boldsymbol{\theta}_m \leq \boldsymbol{\theta}_f \leq \boldsymbol{\theta}_M \tag{4.6}$$

The above equations are now explained:

The objective function 4.2 represents a the minimum displacement of the joints, and depends on the initial posture of the arm. Also note that in this function, λ_k represents the joint expense factors of the arm.

The equality constraint 4.3 represents a non-linear equation imposing that the finger must be in contact with the target object in the final position given by $\boldsymbol{H}(\theta_f) + d_{HO}(\theta_{f,9})\ \hat{z}_7(\theta_f) - \boldsymbol{X}_{tar} = 0$, for the sake of simplicity the function is represented by c_1.

The inequality constraint 4.4 represents a non-linear inequation and depends on how one wants to grasp the object (the desired grip type). About the grip types see on section 6.3 table 6.1 and on section 7.2 table 7.1. For the sake of simplicity, part of the constraint is represented by the function c_2. This constraint compares the frame of the target against the frame of the end-effector under a certain margin (δ);

The inequality constraint 4.5 represents a non-linear inequation of the obstacle pose, of the arm and/or finger angles. For this constraint the points along the arm and hand were previously defined, consisting of spheres with variable radius as it is seen on table 4.1. These points are used in the constraints for checking collisions with the ellipsoids involving the obstacles

and the elliptic cylinder involving the body. For the sake of simplicity, part of the constraint is represented by the function h_f;

The inequality constraint 4.6 restricts the joint values for the final posture between the upper θ_M and lower θ_m values of limits for each joint. It is important to keep in mind that the odd joint limits were inverted for the left arm (see section 3.4)

Note that constraints 4.3, 4.4 and 4.5 require implementation of direct kinematics (see sections 3.4.1 and 3.5.1) in order to calculate the points during the optimization process.

4.2.3 Bounce posture problem - (Pb)

The bounce posture problem is also defined by an objective function that needs to be minimized through optimization processes, while respecting a set of constraints. The main difference between the final and the bounce posture problems is that in the latter the whole movement needs to be calculated for all N_T time instants, from the initial posture to the final posture, passing through the bounce posture to avoid the obstacles in the scenario.

One important aspect of the bounce posture problem is that its dimension slightly varies depending on the task. During grasping movements the **number of joints** n_j that are computed during the optimization process is of nine joints for side and above grips ($n_j = 9$) and ten joints for side grip thumb up ($n_j = 10$) (see sections 6.3, 7.2).

On the other hand, on transport tasks only seven joints ($n_j = 7$) are computed during the optimization process. This was decided because there is no need to keep on planning the grasping of the object once it has been already grasped and is considered part of the arm. Accordingly, new points are generated in order to prevent that the object grasped collides with obstacles.

The bounce posture optimization problem (**Pb**) has the following objective

function and set of constraints, in which $\theta_{b,k}$, where $k = 1, \ldots, n_j$:

$$\mathbf{Pb} \min_{\boldsymbol{\theta}_b \in \mathbb{R}^{n_j}} \quad \sum_{k=1}^{n_j} \lambda_k \left(\theta_{0,k} - \theta_{b,k}\right)^2, \lambda_k \geq 0 \tag{4.7}$$

$$\text{s.t.} \quad \boldsymbol{\theta}_m \leq \boldsymbol{\theta}(t_i, \boldsymbol{\theta}_f, \boldsymbol{\theta}_b) \leq \boldsymbol{\theta}_M \tag{4.8}$$

$$\underline{\boldsymbol{h}}_b(\boldsymbol{\theta}(t_i, \boldsymbol{\theta}_f, \boldsymbol{\theta}_b)) \leq \mathbf{0} \tag{4.9}$$

$$\overline{\boldsymbol{h}}_b(\boldsymbol{\theta}(t_i, \boldsymbol{\theta}_f, \boldsymbol{\theta}_b), \epsilon(t_i)) \leq \mathbf{0}, \tag{4.10}$$

$$\boldsymbol{\theta}_m \leq \boldsymbol{\theta}_b \leq \boldsymbol{\theta}_M \tag{4.11}$$

$$t_i = 0, \ldots, T$$

The above equations are now explained:

The objective function 4.2 represents a the minimum displacement of the joints, and depends on the initial posture of the arm. Also note that in this function, λ_k represents the joint expense factors of the arm and hand.

The inequality constraint (see ineq. 4.8) restricts the joint values for all the iterations i between the upper θ_M and lower θ_m values of limits for each joint. Note the inversion of joint limits for the left arm (see section 3.4).

The inequality constraint (see 4.9) represents a non-linear inequation of the obstacle pose, of the arm and/or finger angles. It also uses points to check for collisions with the ellipsoids involving the obstacles and the elliptic cylinder involving the body for every iteration. For the sake of simplicity, part of the constraint is represented by the function $\underline{h}_b$;

The inequality constraint (see 4.10) represents a non-linear inequation of the obstacle pose, of the arm and/or finger angles. In this constraint the points are used to check for collisions with the ellipsoid involving the target. For the sake of simplicity, part of the constraint is represented by the function $\overline{h}_b$. Note that $\epsilon(t_i)$ is a function of time representing that the clearance distance gets smaller as it gets closer to the target;

The inequality constraint (see 4.11) restricts the joint values for the bounce posture between the upper θ_M and lower θ_m values of limits for each joint.

Note that constraints 4.9, 4.10 also require the use of direct kinematics (see sections 3.4.1 and 3.5.1). The constraint 4.10 is also part of the definition of the problem once during the grasping moment the hand must not collide against the target.

4.3 Optimization tools

Modeling languages are tools that ease and accelerate the decision making process based on the optimization paradigm. In the present days there is a large number of optimization solvers and resources, for solving non-linear problems of large dimensions, that are available online. One of the most complete is the NEOS Server for Optimization `http://www-neos.msc.anl.gov`. For this work, the modeling language used to model the problems was AMPL, and we use the IPOPT solver to find an optimal solution.

4.3.1 A Modeling Programming Language - AMPL

Among the wide variety of optimization tools, the most used modeling language is AMPL (**A** **M**odeling **P**rograming **L**anguage).

This is a modeling language for linear and non-linear optimizations and is the interface where we use the IPOPT solver. It also allows to choose between other solvers and has some options that may improve the solver performance. It is a high-level language that is based on modeling principles and allows for great flexibility.

One of the most notorious advantages of the AMPL is that the user does not need to specify derivatives of the objective and constraints functions. In fact AMPL is able to abstract mathematical models in a compact and perceptible algebraic notation (see fig 4.4).

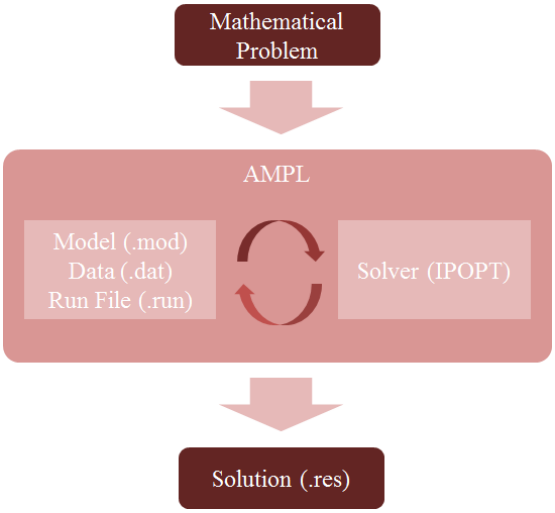


Figure 4.4: AMPL is able to model a mathematical problem into a .mod file a .dat file and runs the simulation through a .run file. A solution is found based on the output of the solver (e.g. IPOPT).

For more information about AMPL see `http://www.ampl.com/`.

4.3.2 Interior-point Optimizer

It has been shown in section 4.3.1 the modelling language used to write the non-linear optimization problem. To solve the non-linear optimization problem ***IPOPT*** was used, which is a solver that implements an *Interior-Point Filter Line search* method.

IPOPT (**I**nterior **P**oint **OPT**imizer) is an open-source software package for large-scale nonlinear optimization. It was designed to find local solutions of mathematical optimization problems 4.5. The optimal solution is obtained by evaluating and computing solutions of a sequence of barrier problems for a decreasing sequence of barrier parameters converging to zero. More information may be obtained on the project website `http://projects.coin-or.org/Ipopt`, see also figure 4.5. It is important to note that there are other solvers for linear and non-linear optimization like **KNITRO**, **SNOPT** and **LOQO**, however none of these were tested in this project. The main reason is due to the fact that in

the previous version of **ARoS** in which this robotic platform performed unimanual tasks, the solvers KNITRO, SNOPT and LOQO were tested and presented higher computational times in most of the purposed tasks (see (Costa e Silva et al., 2011)). Accordingly some of the solvers can not find solutions for several tasks. Also IPOPT is a free software, contrary to the above presented solvers.

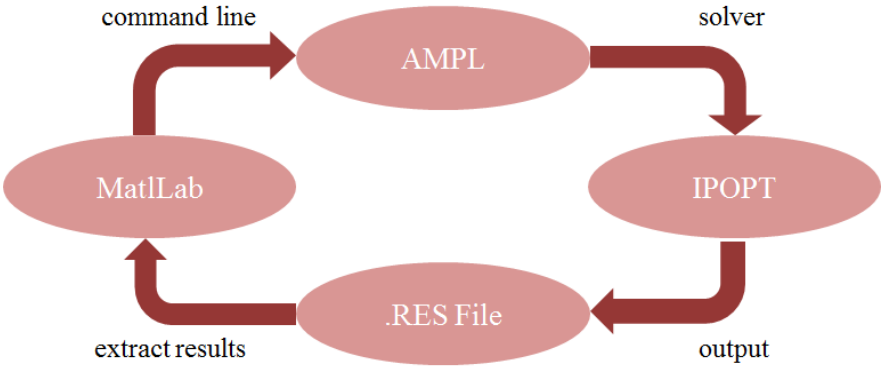


Figure 4.5: IPOPT is used in combination with AMPL in order to solve the optimization problems (Pa and Pb). AMPL is called by Matlab® through command line, abstracts the model and send the problem to be solved by IPOPT. The solution is sent to a .res file which is then read by the Matlab®.

Chapter 5

The Joint Construction Scenarios

In this chapter it will be described two physical scenarios in which the simulations and real tests with **ARoS** in joint constructions tasks, were conducted. The objects that will be grasped and manipulated by both **ARoS** and the user are initially distributed on a table, in both human's and robot's workspace.

5.1 *Toy-Vehicle* Construction Scenario

In this scenario (see fig. 5.1) the following objects placed above a black table may be seen:

- One round base (with 4 holes to insert columns);
- Four columns (yellow and green, yellow and blue, yellow and red, yellow and magenta);
- Two nuts (magenta);
- Two wheels (green).

Figure 5.1, also shows that every object is placed between the human (on the right side of the picture) and the robot (on the left side). The base has a 45

Figure 5.1: Overview of the *Toy-vehicle* construction scenario, unmounted.

degrees rotation in the $\widehat{\mathbf{z}}$ external axis (see fig. 3.2b). It is important to observe that the workspace of both human and robot is limited, which means that neither the robot nor the human are capable of reaching or grasping every object placed on the black table. Consequently cooperation is required in order to complete all the sub-tasks and construct the *Toy Vehicle*. In the robot's workspace there are two magenta nuts placed on its right side and the one column (column 3-yellow and magenta) placed on its left side. All the other objects, that is, columns 1 (yellow and green), column 2 (yellow and red), column 4 (yellow and blue) and both wheels are in the human's workspace. It is also visible that due to the position of the base, **ARoS** is not able to place every object in its final place, **ARoS** can only mount one nut, one wheel and one column, all the other objects must be inserted by the human partner.

In order to simulate and test the toy-vehicle construction scenario, all the objects were designed in SolidWorks® (2012), and a simulator environment in Matlab® (2012b) was also developed. This environment in which the objects, the **ARoS** and the table are visible is shown below on figure 5.2:

This simulator allowed to rapidly test changes in the movement planning for both unimanual and bimanual tasks, including optimization of movements. As

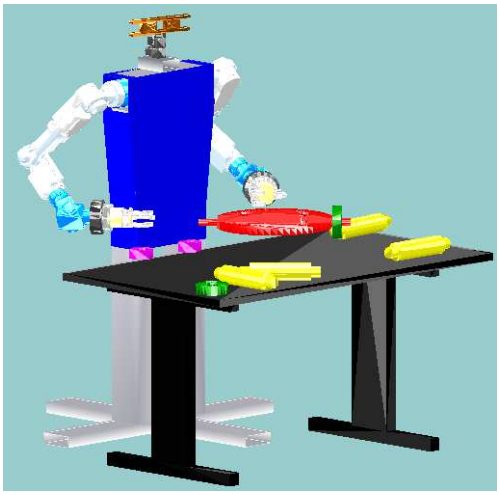


Figure 5.2: Simulation environment of the *Toy-vehicle construction scenario*.

will be seen later, all the cooperative tasks were also simulated through this simulator, taking into account how the robot and the human grasp objects once it may indicate what is the intention beneath.

At end it is expected to have a structure mounted with all 4 columns placed over the circular base, both wheels and nuts placed at the black axles as it is visible on figure 5.3.

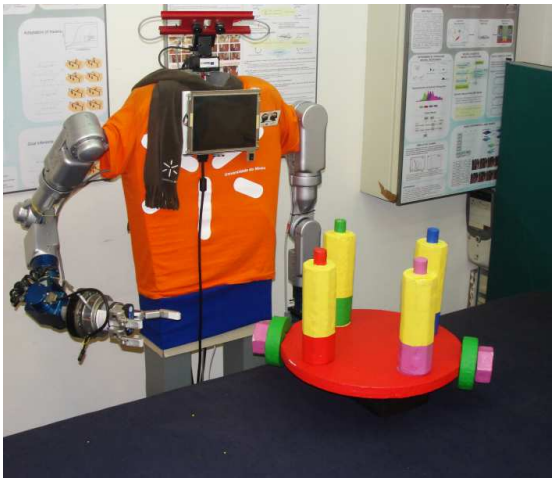


Figure 5.3: Overview of the *Toy-vehicle construction scenario*, mounted.

5.2 Space-Station scenario

In the space station (see fig 5.4) the following objects placed above a black table can be observed:

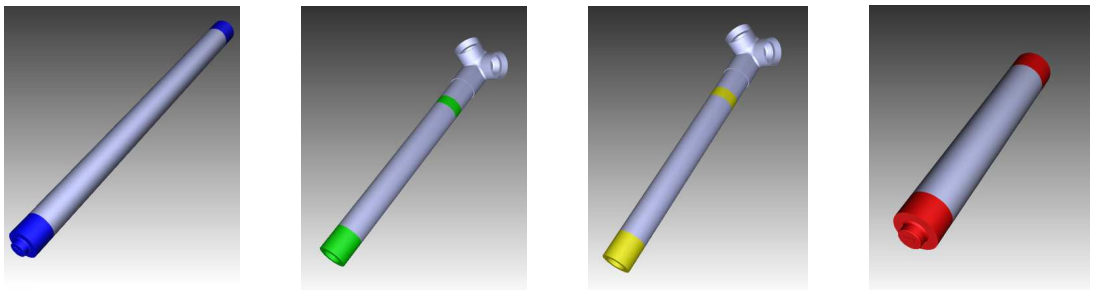
- One rectangular base (with 2 pairs of holes [2 yellow holes and 2 green holes], near its vertices, to insert columns);
- Four base columns (two columns with yellow strips and two columns with green strips);
- Two small columns (with red strips);
- Two big columns (with blue strips).

Figure 5.4: Overview of the *Space-station construction scenario*, unmounted.

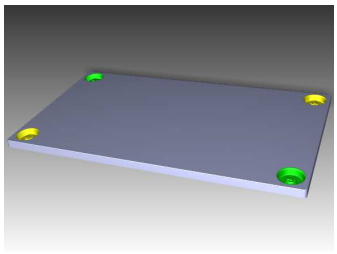
On figure 5.4, one may observe all the columns and the base that are placed on a table between the human (on the right side of the picture) and the robot (on the left side of the picture). The rectangular base is placed perpendicularly to the robot. Once again, identically to the *toy-vehicle* scenario, not every object is directly accessible to both human and robot which means that having the intent to complete the task both actors are required to share tasks and cooperate. This means that despite the initial position of the columns, **ARoS** or the human user are not able to mount every column in its place for itself/himself. Due

to workspace limitations the robotic arms are only able to reach and place both green and yellow columns at the holes placed on its side despite being able to help the human by handing him the columns to be further placed. The same happens to the human partner who can help the robot by handing him the columns but cannot place them in the holes placed on the **ARoS** side.

In order to simulate (see fig. 5.2) and test the *space-station* scenario, it was necessary to design all the object by using SolidWorks® 2012, as may be seen below on figure 5.5:



(a) Blue column. (b) Green column. (c) Yellow column. (d) Red column.



(e) Base.

Figure 5.5: *Space-Station* scenario, developed using SolidWorks® 2012.

A simulator environment in Matlab® (2012b) was developed, where the objects used in the Space Station scenario the **ARoS** and the table may be seen in figure 5.6. With this simulator it was possible to test changes in the movement planning for bimanual tasks. Furthermore, it allowed debugging and testing optimizations of movements and improve human likeness. As will be seen later, all the cooperative tasks, between the human and the robot, in the space station scenario were also simulated through this simulator.

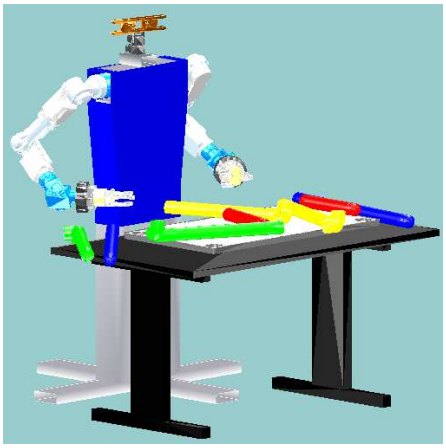
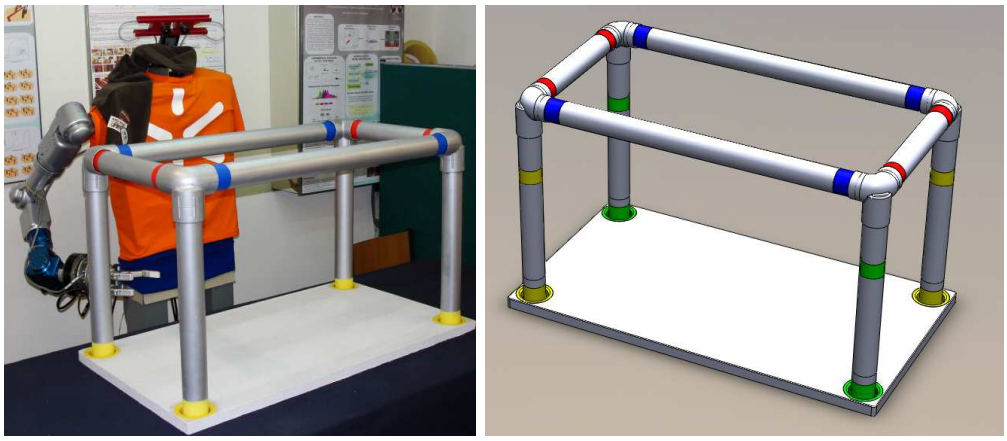


Figure 5.6: Overview of the *Space-station construction scenario*, unmounted.

Finally, this scenario is supposed to show that **ARoS** is capable of perform bimanual tasks and to cooperate with the user with human-likeness. The ultimate goal is to have the structure completely mounted as seen on figure 5.7:



(a) Space station Physical Model. (b) *Space-station* developed using SolidWorks® 2012.

Figure 5.7: *Space-Station* scenario fully mounted (as seen on the left figure) and developed using SolidWorks® 2012 (as seen on the right figure).

Note that, as will be explained later on chapter 7, section 7.3.1 , it is not part of the project's aims to have this structure completely mounted. As mentioned previously, with this scenario it is intended to demonstrate bimanual cooperative

capabilities of **ARoS** with smooth and natural movements.

5.3 Common and different points between *Toy-Vehicle* construction scenario and *Space-Station* construction scenario

There are some common points to be considered between both scenarios:

- First of all, both scenarios have a high number of objects (nine) cluttering the workspace. With a high number of objects it is possible to solve more challenging problems and it becomes more similar to what can be found in human-centered scenarios;
- Every task in each construction scenario is hampered by the presence of obstacles that the manipulator must avoid;
- Having real-time movement planning that reacts to the human user and dynamical changes in its workspace (e.g. objects changing its position), each scenario follows a sequence of sub-tasks (explained later on chapters 7 and 6). Both human and robot must follow the defined sequences in order to complete the purposed tasks;
- Before the beginning of the task, every object is placed in the workspace of both robot and human user, this urges for cooperation since theirs workspace is limited. Accordingly the human or the robot may need to reach or grasp one object that is not in his/its workspace;
- Both human and robot are able to assemble the scenario in his/her/its workspace;
- Despite having only one possible final state does not mean that there is only one logical order to construct the scenario (there are several possible

ways to achieve construction plan);

- For both scenarios **ARoS** bimanual movements consists of Asymmetrically Coordinated tasks (see section 1.1.6) which are logically dependent cooperative (see section 1.1.6), accordingly to Guiard (1987); Zacharias et al. (2011).

The main differences of both scenarios are now described:

- The *Space-Station* scenario has bigger objects. Accordingly to this, the cooperation or bimanual manipulation is needed during almost every task given the size of the object and the physical characteristics of the structure. In the *Toy-Vehicle* construction scenario all objects are smaller, which means that tasks are more easily conducted using just one arm. However, the cooperation is still essential to pass objects that are outside the Human or robot workspaces;

- The shape of the objects is more similar in the *Space-Station*. In this scenario every object has a cylindrical shape, guiding the vision and the cognitive control systems into more challenging problems. However it becomes easier to grasp the objects once it requires less grip types.

Before explaining the movements observed in both scenarios sequences (see section 7.3.1 and 6.4.1) one should understand the movement primitives, meaning the elementary movements ordered by the cognitive architecture (see, (Bicho et al., 2011a)). Understanding what these primitive movements consist of, will ease the understanding of the sequences implemented in both scenarios.

5.4 Elementary Movements

The elementary movements, which are the basis of the implemented sequences in *Toy-Vehicle* and *Space-Station* scenarios are the following:

Ask for object - Covers movements of arm and hand in which the arm moves from its home/initial position towards a customizable position near the human but still within its workspace. After reaching this virtual point near the human, the fingers open to a fixed value and when receiving the object, completely close around it, in order to firmly hold it;

Reach to grasp an object at the table or at the human hand - Covers movements of the arm and hand in which the arm moves from its home position towards the position of the object to grasp. It is important to understand that the grip type and the fingers preshape to grasp the object are given by the cognitive architecture (see, (Bicho et al., 2011a)). In the case of grasping the object from the human hand, the robot does not need to wait for the human to start his movement;

Hand over an object - Covers movements of the arm, it is important that the fingers do not move once they are holding an object to hand over. The arm goes from a current posture (where the object was grasped) to a position near the final posture obtained at the movement *"Ask for Object"*. This means that the manipulator is moving in direction to its boundaries in order to hand over an object to the human. **ARoS** decides it must hand over an object and starts the move towards the human, despite he/she is already asking for the object or not, with the objective to make a faster and smoother interaction. If the human is already asking for the object, the **ARoS** will try to transport the object to a position as close to the human as possible;

Insert an Object - Covers movements of the arm, it is important that the fingers do not move once they are holding an object to insert in the base. In fact, this movement is composed of two sub-movements: **1)** from the current position to a location close to the final position, note that the location is behind the final position in the case of nuts and wheels and above the

final position to columns); **2)** from the position explained in 1) until the final position;

Release back-off and return to home - Covers movements of the arm and hand. This movement is composed of two sub-movements: **1)** in which the hand of the manipulator release the object (the movement is accomplished by spreading the fingers); **2)** after releasing a given object the manipulator must back-off to a near pre-defined position 1) and then return to its home position.

It is possible to subdivide these movements into two different categories: Simple movements that are composed of one main movement and complex movements composed of two movements:

- **Simple movements -** Ask for object, Reach to grasp an object placed at the table or in the human hand, Hand over an object;
- **Complex movements -** Insert an object, Release back-off and return to home.

Note that there are alternative categorizations that can be used to group different kinds of movements. In this work the movements were categorized giving the number of degrees of freedom that are being controlled at the same time in the movement planning. The main reason is that there is no need to calculate all the eleven (seven DOF's from the arm, three from the fingers and one to spread two fingers related to each other) joints during the movement planning of each movement (see section 4.2.3).

5.4.1 Home position

This position is the one in which the robot starts and finishes its sequence of movements, also it is the position that **ARoS** takes between two independent movements. Before starting a new sequence of movements and after ending the

previous sequence of movements, **ARoS** goes to the Home Position 5.8. This behavior is similar to what is observed in humans. In this position the robots hand is placed slightly ahead of its body and slightly in its right side for the right arm and slightly in its left side for the left arm. The fingers are opened but not completely, adopting a more natural position. Note that the home position refers to the resting position and not to the position in which the arm joints have a null value.

(a) Home position - Right Arm. (b) Home position - Both Arms (c) Home position - Left Arm

Figure 5.8: Three different perspectives of the **ARoS** home position in which both arms are visible - Left, center and right perspectives.

Chapter 6

ARoS Movements - *Toy-Vehicle* Information, Sequence of Construction and Results

6.1 Importance of handing over or asking for an object

Tasks involving handing over (see fig. 6.1) or asking for an object are necessary to keep a well established social human-robot interaction. This is exacerbated if there are objects the robot needs to grasp which are in the workspace of the human, being the case observed in *Toy-Vehicle Construction*. Considering this situation, movements of *Hand-Over* and *Ask for object* (see, (Silva, 2011)) have been optimized in order to be calculated fast enough to be used in real-time situations. For more information about the movements check section 5.4.

6.2 Toy-Vehicle Workspace

In the Toy-Vehicle scenario, both human user and **ARoS** have a well defined workspace in which they are allowed/able to grasp an object. The workspace is

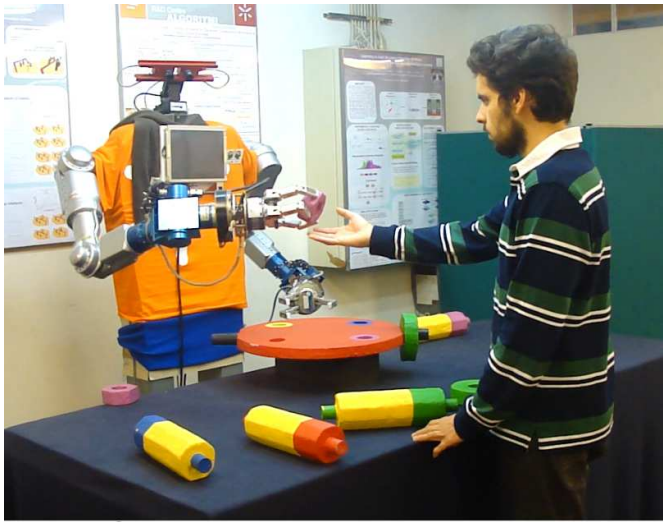


Figure 6.1: Enabling the possibility to have shared tasks between the human and the robot, increases the range of possibly tasks to be accomplished. In this figure it is visible **ARoS** handing over a nut to the human user.

defined by a imaginary line (red line in figure 6.2) that divides the black table in approximately half of the black table for each entity. Nevertheless there are some rules that may relax the idea of a insurmountable line:

- The human user cannot start a task inside the workspace of the robot, however he/she is able to help **ARoS** performing a task inside its workspace as a secondary entity;

- **ARoS** is not allowed by any means to enter inside the human workspace, nonetheless in movements that include handing over an objects to the human, the object grabbed by the fingers may enter the human workspace.

Finally limiting workspaces (by physical needs or by definition), also promotes the sharing of tasks once not every task in the scenario is reachable for both human and **ARoS**.

Figure 6.2: Human and robot workspaces, on the left and right side of the figure respectively.

6.3 Grip types

Different movements and different intentions require different kinds of grips, taking into account the position of the object in the scenario. Being supported by the work developed by Napier (see (Napier, 1956), all the grip types used in this project were *precision grip*. This kind of grip consist on grasp using only the fingers against which the objects are pressed using the opposing thumb (the palm of the hand is not used). It is possible to subdivide this type of grip considering the relative orientation of the robot's hands and object (see fig. 6.3).

Figure 6.3: Relation between the orientation of the object and the hand of the robot.

Figure 6.3 helps to understand the relation between the orientation of the hand and the object, in this grip type the hand is perpendicular to the height of the object.

In table 6.1 are presented all the grip types used in this specific sequence, as well as the axis and angles of the robot hand for each grip type:

Table 6.1: Relation between Grip Type, Rotation of the hand, Angles and the Orientation of the Object. As may be seen in the 2^{nd} column there are two different orientations the hand can call upon in order to grasp objects that were placed on the table with different orientations (last column).

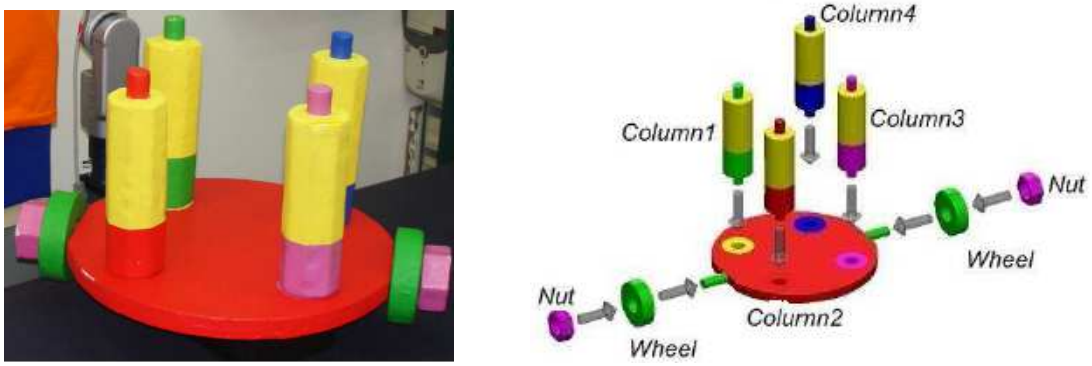
Grip type (precision grasp)	Rotation Axis	Angle (deg)	Orientation for the new frame	Grip code
Side grip - Thumb Left	$\widehat{\mathbf{x}}_{tar}$	0	$\widehat{\mathbf{z}}_{tar}$	111
Side grip - Thumb Up	$\widehat{\mathbf{x}}_{tar}$	90	$\widehat{\mathbf{y}}_{tar}$	113
Side grip - Above	$\widehat{\mathbf{y}}_{tar}$	-90	$\widehat{\mathbf{x}}_{tar}$	121

6.4 *Toy-Vehicle* brief explanation and sequence of construction

As mentioned previously, the development of a system that is able to actively cooperate with the human user is one important aim of this project. Accordingly, it is of outmost importance to test situations in which this cooperation is being monitored.

Considering this goal, an explanation of one of the tasks in which the movement planning was tested is required. The task consists of the assembly of a toy from objects that were initially distributed on a table (see also, (Bicho et al., 2011b; Silva, 2011)). The toy (see fig. 6.4a) consists of a round base with two axles in which two wheels (wheels of the vehicle) must be inserted and two nuts

to hold the wheels in their places. Above the base, four columns must be inserted into their respective holes (see fig. 6.4b):



(a) *Toy-vehicle* in its final form. (b) Explanation of how to mount the toy vehicle

Figure 6.4: The left figure shows the toy completely mounted while the right figure show how the Toy-Vehicle should be mounted.

6.4.1 Sequence of movements used for the construction

After the previous explanation, it is now possible to present in detail the sequence of movements of the robot in order to mount the *Toy-Vehicle* for this distribution of objects:

Note that this scenario starts with a small-wheel already inserted in the axle that is in the human user workspace.

Step 1 - Reach Nut with Right Hand, Above-Grip to the Human (RN_Ra_AG_H) - Pick the nut placed in its workspace (see fig. 6.5b), (the nut closer to its body), with the right hand and passes above the black table to the human-user, approaching to the boundaries of its workspace.

After handing over the nut to the human and while the arm is returning to its home position, the human user inserts the nut (see fig. 6.5c). Note that is impossible for the robot to pass to the next step while the human

has not yet finished his action. Afterwards the robotic arm comes back to its home position;

Step 2 - ARoS requesting to human the wheel with its right hand (RhtoH_EW_Rh) - In this step the robot must move its right arm towards the human's workspace in order to request the wheel (see fig. 6.5d) which is in the human workspace and cannot be reached by any **ARoS** arms;

Step 3 - Reach Wheel from Human Above Grip and Insert with right hand (RwhH_AG_Rh_I) - Grasp the wheel from the human hand (see fig. 6.5e) with the right arm. Synchronization between human hand and the manipulator is not required. The manipulator may travel towards the human space before the human start the movement to deliver the wheel or vice-versa. After grasping the small wheel with the grip type thumb above, the manipulator must move towards the axle of the base in its workspace and finally insert the wheel in its final place (see fig. 6.5f). Afterwards the robotic arm comes back to its home position;

Step 4 - Reach Nut, Side Grip thumb up and Insert with right arm (RN_Rh_SG_I) - In this step the right arm has to find an acceptable path between its home position and the nut placed in the right side of its workspace. After getting close to the nut the hand must grasp it using a side grip configuration (see fig. 6.5g).

Subsequently, after grasping the nut with its right hand, **ARoS** must insert the object in the axle of the base (see fig. 6.5h) leaning against the wheel that was already inserted during the step 3. After placing the nut in its final position, the right arm must return to its home position;

Step 5 - Request to human Column 1 with the right arm (RhtoH_ECol1_Ra) - In this step the robot must move its right arm towards the human workspace in order to request the Column 1 (green and yellow), because this column needs to be inserted in the **ARoS** workspace

but the column is lying in the human workspace, out of its reach (see fig. 6.5i);

Step 6 - Reach Column 1 from Human, Above Grip and Insert with the right hand (RCol1hH_AG_I_Rh) - During this sub-task the right arm must grasp the column 1 (see fig. 6.5j) from the human hand moving its right arm from the request position towards the position where the human user placed the column 1. Similarly to what happens during the step 3 the arm may start its movement towards the human workspace before the human user starts his/her movement to handover the column.

After the arm of the robot is in its final position the hand must grasp the column using a side grip. Afterwards, having grasped the column, the manipulator must move towards the local of placement, inserting the column in the respective hole in the base (see fig. 6.5k). After the column is inserted, the right arm return to its home position;

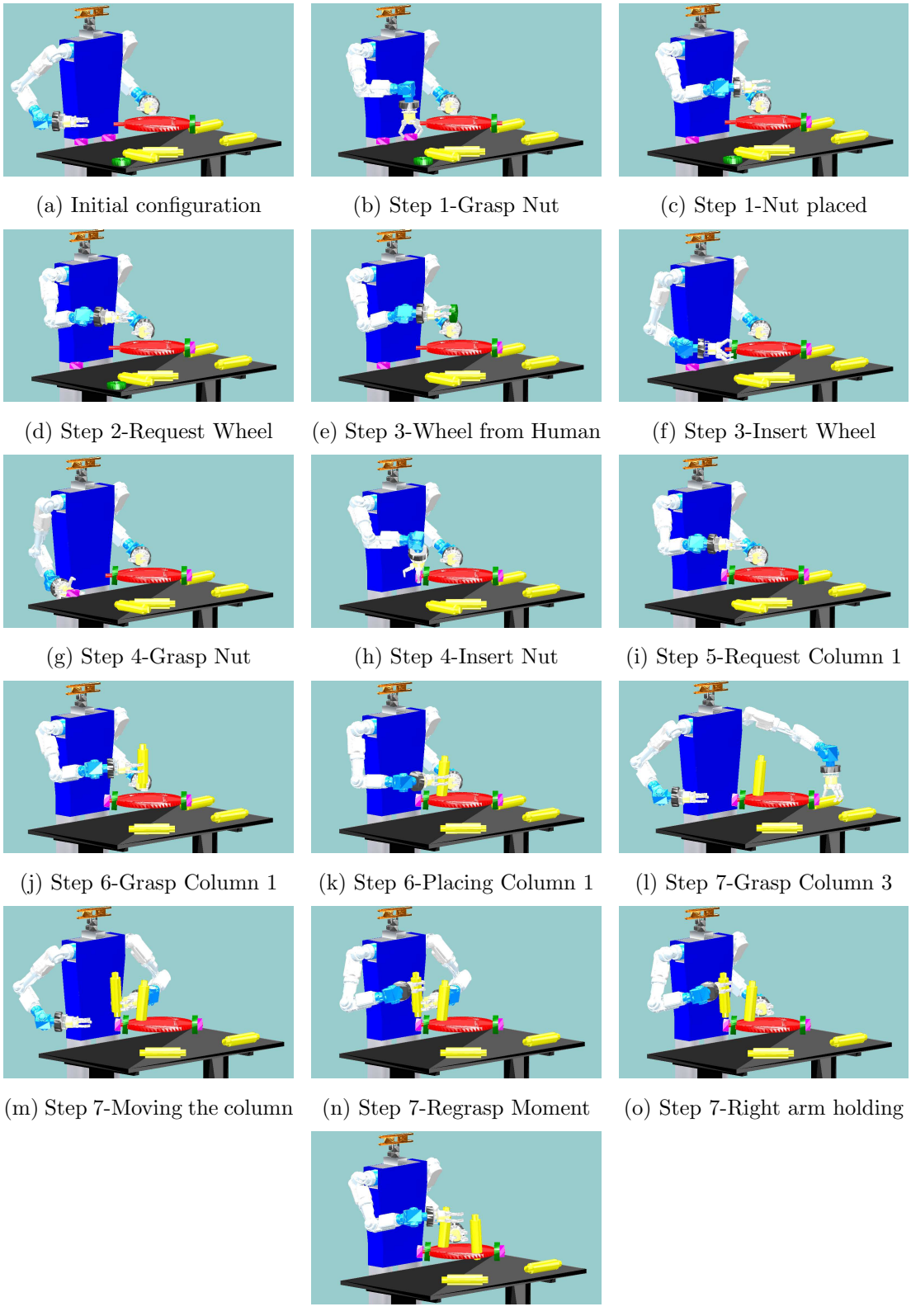
Step 7 - Reach Column 3, Side Grip with the Left Hand pass Right hand and using Side Grip pass to Human (RCol3_SG_Lh_SG_Rh) - In this step the left arm grasps the Column 3 (magenta-yellow, see fig: 6.4b) using a side grip configuration which has the thumb to the right side of the object (see fig. 6.5l). Note that the Column 3 is placed at the left side of the robot's workspace, and that is why **ARoS** uses the left arm. During the grasping movement, the fingers of the robot grab the left tip (in the **ARoS** perspective) of the object.

After grasping the object, the right arm moves towards the center of the scenario near **ARoS** body but lifting itself a little to avoid coliding with the base. The objective of this movement is to make possible the regrasping by the other arm. Once the left arm reaches its final position (see fig. 6.5m), at the regrasping posture, the right arm starts its movement towards the object that is being grasped by the left arm.

The right arm has to grasp the column nearest to the right tip (see fig. 6.5n) considering the **ARoS** perspective. In order to ease the regrasping movement the right arm cannot grasp the object in the middle of the column that would preclude the grasping by the right arm given that it would not have enough space to grab the object.

Notice that the right arm also grasps the column with a side grip configuration (with the thumb in the right side of the object). After having grasped the column with the right arm, the left arm releases the object and returns to its home position (see fig. 6.5o). Now the column is being grasped only by the right arm, and the task is to handover the column to the human who will insert it in the base. In order to accomplish this task, the robot moves towards the boundaries of its workspace to ease the grasping by the human user. After the robot finishes this movement, the human user grasps the column 3 and insert it (see fig. 6.5p).

One should note that the robot is not able to place the column 3 by itself into the base, since the correspondent hole is out of the reach. **ARoS** finishes the sequence of movements by returning to its home position.



(a) Initial configuration (b) Step 1-Grasp Nut (c) Step 1-Nut placed

(d) Step 2-Request Wheel (e) Step 3-Wheel from Human (f) Step 3-Insert Wheel

(g) Step 4-Grasp Nut (h) Step 4-Insert Nut (i) Step 5-Request Column 1

(j) Step 6-Grasp Column 1 (k) Step 6-Placing Column 1 (l) Step 7-Grasp Column 3

(m) Step 7-Moving the column (n) Step 7-Regrasp Moment (o) Step 7-Right arm holding

(p) Step 7-Deliver Column 3

Figure 6.5: Sequence of movements obtained during the simulated construction of the *Toy-Vehicle*.

As shown in figure 6.5 the sequence to construct the *Toy-Vehicle* is composed of seven main steps in which steps one, two, three, four and five were composed of two sub-steps and step seven is the most complex movement involving five sub steps (since it includes movements of both right and left arms). In figure 6.6 the seven steps required to construct the *Toy-Vehicle* are shown, that focus on the grasped and handled objects, contrary to figure 6.5.

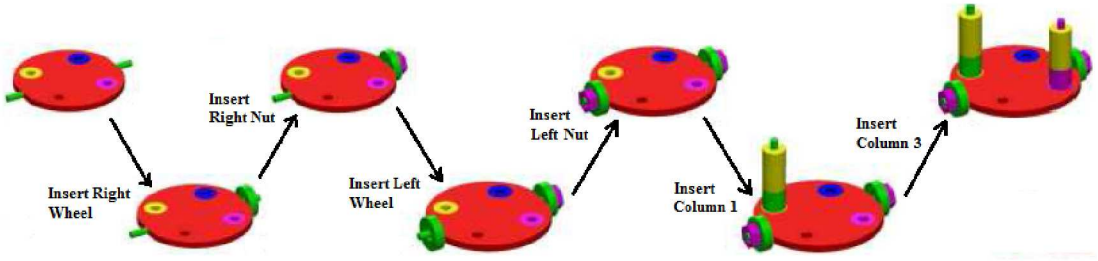


Figure 6.6: Construction sequence of the *Toy-Vehicle* scenario from the first until the last step during the **ARoS**-Human interaction.

6.5 Results for unimanual and bimanual *Toy-Vehicle* construction scenarios

This section will focus on the analysis of the movement sequence implemented in the Toy-Vehicle joint construction generated by the movement planning in a human-robot joint construction task. Descriptions of the sequence of movements are presented in table 6.2 and table 6.5. Results will be shown in tables presented in this section.

All optimization problems, **P#a** (final postures) and **P#b** (bounce postures), were coded in AMPL modeling language and solved using IPOPT 3.11. The numerical results were obtained using an Intel(R) Core(TM) i5 M460@2.53GHz processor running Windows 7 64-bits with a ATI Mobility Radeon HD 5650 video card and 4GB of Ram Memory.

6.5.1 Results in the unimanual Toy-Vehicle sequence, presented at the *"ICNAAM 2013"*

A smaller sequence than the one described in section 6.4.1, was submitted to the *International Conference of Numerical Analysis and Applied Mathematics* (ICNAAM 2013) in the symposium related to *Numerical Optimization and Applications* (NOA 2013). The paper was entitled *"Generating human-like movements on an anthropomorphic robot using an interior point method"* and was accepted for "AIP Conference Proceedings" for presentation at ICNAAM 2013 in Rhodes-Greece during 21 to 27 of September 2013.

Before implementing the complete sequence seen on figure 6.5 the optimization parameters were tuned in a shorter sequence of movements. The main purpose was to verify how the Matlab® simulator perform the arm movements. This small sequence was also important in order to conduct some adjustments and improvements in the mathematical models and parameters. As will be seen further ahead, the sequence of movements described in this section consists only of unimanual movements, using just the right arm.

6.5.1.1 Problems description

More specifically, **ARoS'** movements for this sequence are the following:

- **P1** requests an object from the human;
- **P2** grasps the wheel that the human hands it;
- **P3** transports and inserts the wheel in the base;
- **P4** returns its arm to the home position.

Table 6.2 presents details on each of these movements.

Table 6.2: Description of the shorter version of the Toy-Vehicle sequence presented at NOA 2013. Here is visible both **Pa** and **Pb** sub-problems of this sequence in which Pa is for final posture selection problems (final value of all the calculated joints) and Pb is for bounce posture selection problems (sequence of postures calculated to reach the previously obtained final posture)

Prob.	Movement	Target object	Objects in the workspace	Pa	Pb
1	Request object	-	Table, base, wheel and column	-	**P1b**
2	Reach-to-grasp wheel	Wheel	Table, base and column	**P2a**	**P2b**
3	Transport and insert wheel	- -	Table, base and column	**P3a**$_1$+**P3a**$_2$	**P3b**
4	Return home	-	Table, base, column and wheel	**P4a**	**P4b**

6.5.1.2 Parameters used in this sequence

For the implementation of the short sequence of the Toy Vehicle scenario, several parameters have been set. These parameters are the following:

- Movement duration is $T = 1$. Note that this is a unitary value and after having the trajectory planned, the duration is adapted in order to limit the velocity of the joints;
- Precision of the orientation of the end-effector $\delta = 1e-2$ (for **P2a** $\delta = 1e-3$);
- Joint expense factors have a default value of $\lambda_k = 1$ for $k = 1, \ldots, n_j$. This means that all joints contribute equally to the movement, (i.e. every joint has the same associated cost);
- IPOPT was run with the default options: Tol $= 1.0e-6$ and an acceptable tol $= 10.0e-6$. Those are related to error margin accepted for the optimality conditions of the problem in which a solution is considered opti-

mal. Second order derivatives were approximated using a limited-memory method, named Broyden-Fletcher-Goldfarb-Shanno (L-BFGS);

- AMPL presolve was set to off in order to maintain the original equations. The presolve setting simplifies the equations of the model, but it was observed that the new equations obtained did not produced an optimal solution for the expected behavior.

6.5.1.3 Posture problem results

Tables 6.3 and 6.4 show the numerical results of this sequence. In these tables the following variables are presented:

- **Number of variables,** N directly affects the dimension of a problem and is related to the number of joints being calculated;
- **Number of equality constraints,** M_{eq}, is related to constraints of the problem that are defined as equalities;
- **Number of inequality constraints,** M_{ineq}, is related to constraints of the problem that are defined as inequalities;
- **Number of non-zero elements in equality constraint Jacobian,** n_{eq}, is directly dependent of the equality constraints and affects the dimension of the problem;
- **Number of non-zero elements in inequality constraint Jacobian,** n_{ineq}, is directly dependent of the inequality constraints and affects the dimension of the problem;
- **Objective function value (Obj)**, is the minimum value found for the objective function, while respecting the constraints of the problem, which means, having an acceptable value that verify the optimality conditions of the problem. The smaller optimality value the minor becomes the constraint violation obtained. It is considered a requirement that the value

of the constraint violation is below the tolerance defined in the IPOPT's parameters;

- **Computational time in seconds, CPU,** is affected by the dimension of the problem and the difficulty of minimizing the objective function subject to the set constraints on its variables. This may be related to the complexity of the workspace and more or less computationally demanding position of the target to be grasped.

The results related to the final posture selection problem (Pa) are presented in table 6.3: The results related to the bounce posture selection problem (Pb)

	P2a	**P3a$_1$**	**P3a$_2$**	**P4a**
N	7	7	7	7
M_{eq}	3	3	3	3
M_{ineq}	75	105	126	126
n_{eq}	17	17	17	17
n_{ineq}	439	649	778	778
Obj	0.2465	0.7525	0.0495	0.0719
CPU	0.156	0.203	0.171	0.312

Table 6.3: Numerical results for **Pa** subproblems.

are presented in table 6.4:

	P1b	**P2b**	**P3b**	**P4b**
N	9	9	7	9
N_T	10	10	20	10
M_{ineq}	711	906	786	531
n_{ineq}	3660	4900	2920	2425
Obj	3.663e-013	1.338e-02	1.548e+0	1.271e-015
CPU	0.858	0.749	1.123	0.764

Table 6.4: Numerical results for **Pb** subproblems.

IPOPT managed to easily find an optimal solution for all final posture selection problems in less than 0.320 seconds. An optimal solution was found for all bounce posture selection problems in less than 1.2 seconds. All the posture selection problems are of medium dimension considering the number of constraints

(M_{eq}, M_{ineq}) and variables (N) which is between 100 and 1000 except for P2a which is of small dimension. The movement of inserting and placing the wheel on the base was the one that presented the greater risk of collision with the surrounding obstacles. Therefore it was not surprising that **P3b** was the most challenging subproblem for the IPOPT solver, as can be seen by the CPU time in table 6.4. Also for this movement the bounce component, responsible for avoiding collision, was greater than for all the other movements. This is expressed by the value of the objective function.

Note that for the bounce posture selection problems some tuning was necessary in the definition of the clearance distance $\epsilon(t)$.

6.5.2 Results in the bimanual Toy-Vehicle construction scenario

Here the results of the complete sequence of movements as described in section 6.4.1 are displayed. Having the shorter sequence of the Toy-Vehicle optimized (sequence described in table 6.2), the basic adjustments were performed and it was possible to move on into a more complex sequence of movements. Nevertheless additional tunings had to be conducted on the simulator and in the AMPL's mathematical models with the purpose to endow the movement planning with the required robustness and efficiency to complete complex movements. As will be seen later, the sequence of movements described in this section consists of unimanual and bimanual movements.

6.5.2.1 Importance of bimanual movements for assembling a complex structure

Most of the joint tasks are so common that humans do not even realize that perform interactions in order to ease, optimize and accelerate the flow of a given task. Furthermore in order to achieve more complex constructions, one should note that such tasks might be impossible to complete if no bimanual interaction and cooperation exists during the construction process (e.g. one actor opens the

door while the other transports a stack of books with both hands). In short, simultaneous movements of both arms between the human and the robot bring vantages such as:

- Allows time-efficiency better performance against the same task accomplished with a series of unimanual movements (easily noticeable for complex tasks);
- Possibility to accomplish tasks that require both the human and robot to use two arms.

With these advantages in mind, the goal was to construct a scenario where a task should be developed involving both human and **ARoS** relying on bimanual movements, in order to complete the task.

6.5.2.2 Problems description

More specifically, **ARoS'** movements for this sequence are the following:

- **P1:** Reach to grasp a nut from the black table with the right arm (**1.1**), hand over the nut to the human (**1.2**), then returns the arm to its home position (**1.3**);
- **P2:** Ask for a wheel with the right arm;
- **P3:** Reach to grasp a wheel from the human with the right arm (**3.1**), inserts it in the base (**3.2**) then returns the arm to its home position (**3.3**);
- **P4:** Reach to grasp the other nut from the table with the right arm (**4.1**), inserts it in the base (**4.2**) then returns the arm to its home position (**4.3**);
- **P5:** Ask for column 1 with the right arm;
- **P6:** Reach to grasp the column 1 from the human with the right arm (**6.1**), inserts it in the base (**6.2**) then returns the arm to its home position (**6.3**);
- **P7:** Reach to grasp the column 3 from the table with the left arm (**7.1**), regrasp movement to the right arm (**7.2** and **7.3**), returns the left arm to the home position (**7.4**), hand over column 3 to the human (**7.5**) and finally returns the right arm to its home position (**7.6**).

Table 6.5 presents details on each of these movements.

Table 6.5: Problems description.

Problem	Movement	Target object	Other objects in the workspace	Pa	Pb
1.1	Reach-to-grasp Nut	Nut	Table, base, nut and column	**P1.1a**	**P1.1b**
1.2	Hand Over Nut	Nut	Table, base, nut and column	-	**P1.2b**
1.3	Return To Home	-	Table, base, nut and column	-	**P1.3b**
2	Ask Wheel	-	Table, base, nut and column	-	**P2b**
3.1	Reach-to-grasp Wheel	Wheel	Table, base, nut and column	**P3.1a**	**P3.1b**
3.2	Insert Wheel	-	Table, base, nut and column	**P3.2a**$_1$ + **P3.2a**$_2$	**P3.2b**
3.3	Return To Home	-	Table, base, nut and column	**P3.3a**	**P3.3b**
4.1	Reach-to-grasp Nut	Nut	Table, base, nut and column	**P4.1a**	**P4.1b**
4.2	Insert Nut Nut	-	Table, base, nut and column	-	**P4.2b**
4.3	Return To Home	-	Table, base, nut, wheel, column	-	**P4.3b**
5	Ask Column	-	Table, base, nut, wheel, column	-	**P5b**
6.1	Reach-to-grasp Column	Column	Table, base, nut, wheel, column	**P6.1a**	**P6.1b**
6.2	Insert Column	-	Table, base, nut, wheel, column	**P6.2a**$_1$ + **P6.2a**$_2$	**P6.2b**
6.3	Return To Home	-	Table, base, nut, wheel, 2 columns	**P6.3a**	**P6.3b**
7.1	Reach-to-grasp Column	Column	Table, base, nut, wheel, 2 columns	**P7.1a**	**P7.1b**
7.2	Regrasp left arm	-	Table, base, nut, wheel, 2 columns	-	**P7.2b**
7.3	Regrasp right arm	-	Table, base, nut, wheel, 2 columns	**P7.3a**	**P7.3b**
7.4	Return home left arm	-	Table, base, nut, wheel, 2 columns	-	**P7.4b**
7.5	Handover Column	-	Table, base, nut, wheel, 2 columns	-	**P7.5b**
7.6	Return Home	-	Table, base, nut, wheel, 2 columns	-	**P7.6b**

6.5.2.3 Parameters changed for this sequence

For the implementation of the full sequence related to the Toy-Vehicle scenario the following parameter was changed:

- Precision of orientation of the end-effector $\delta = 1.0e-2$ (for **P2a** $\delta = 1.0e-3$);

6.5.2.4 Posture problem results

From table 6.6 to table 6.11 it is shown both final and bounce posture numerical results of this sequence. In these tables the following variables are presented:

- Number of variables N,
- Number of equality constraints, M_{eq}
- Number of Inequality constraints, M_{ineq}
- Number of non-zero elements in equality constraint Jacobian, n_{eq}
- Number of non-zero elements in inequality constraint Jacobian, n_{ineq}
- Objective function value, Obj
- Computational time in seconds, CPU

The results related to the final posture selection problem are presented in tables 6.6 and 6.7:

Table 6.6: Numerical results for **P1.1a-P4.1a** subproblems

	P1.1a	**P3.1a**	**P3.2a$_1$**	**P3.2a$_2$**	**P3.3a**	**P4.1a**
N	7	7	7	7	7	7
M_{eq}	0	3	0	0	0	0
M_{ineq}	68	60	68	68	68	50
n_{eq}	0	17	0	0	0	0
n_{ineq}	411	352	411	411	411	286
Obj	2.812e+0	2.465e-1	4.856e-1	7.401e-1	7.438e-2	7.877e-1
CPU	0.546	0.125	1.919	2.745	0.749	0.624

Table 6.7: Numerical results for **P6.1a-P7.3a** subproblems.

	P6.1a	**P6.2a$_1$**	**P6.2a$_2$**	**P6.3a**	**P7.1a**	**P7.3a**
N	7	7	7	7	7	7
M_{eq}	0	0	0	0	0	0
M_{ineq}	76	65	65	65	76	76
n_{eq}	0	0	0	0	0	0
n_{ineq}	446	345	345	345	411	446
Obj	5.661e-2	1.779e-1	1.698e-1	1.555e-1	3.963e+0	2.817e+0
CPU	0.187	0.749	4.976	1.014	0.250	0.328

The tables (6.6 and 6.7) show that IPOPT found successfully an optimal solution for all the eleven final posture selection problems, in a reasonable time for Human-robot interaction. The times to obtain the solutions for all problems range between 0.125 seconds and 2.745 seconds. As observable in the table 6.7, the time spent to solve P6.2a$_2$ is almost 5 seconds (4.976 seconds), this time is justified by the illustration 6.9f in which is visible that in order to insert column 1 the manipulator has to get near three close obstacles (nut, wheel and base).

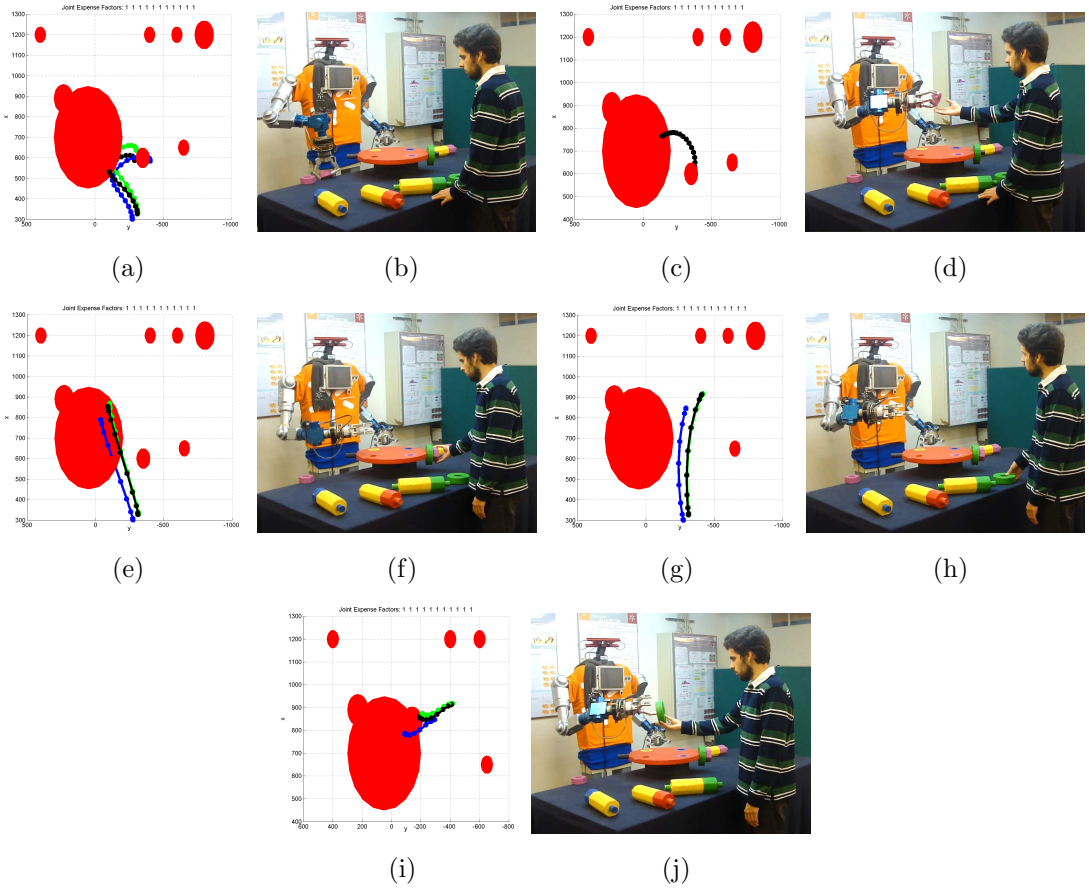
Therefore it becomes a demanding challenge to find an optimal solution to insert the column in its final place, taking in account the nearby obstacles. Every problem seen in tables (6.6 and 6.7) are of small dimension considering the number of constraints (M_{eq}, M_{ineq}) and variables (N) which is less than 100. In some cases (P1.1a and P7.3a) the value of the objective function found is relatively high comparatively to other objective functions of final postures. This is due to the fact that these cases corresponds to grasp an object from the table which is a demanding problem to be solved comparatively to other final posture selection problems. Nevertheless even in this cases the tolerance is respected.

The results related to the bounce posture selection problem are presented in tables 6.8, 6.9, 6.10 and 6.11.

The next table presents the numerical results for bounce postures related to movements from P1.1b to P3.1b, in which the related data is presented and afterwards explained.

Table 6.8: Numerical results for **P1.1b-P3.1b** subproblems

	P1.1b	**P1.2b**	**P1.3b**	**P2b**	**P3.1b**
Arm	Right	Right	Right	Right	Right
N	10	7	9	9	9
N_T	20	20	10	10	10
M_{eq}	0	3	0	0	0
M_{ineq}	1521	639	546	546	906
n_{eq}	0	0	0	0	0
n_{ineq}	8812	2290	2995	2995	4900
Obj	9.367e-02	5.058e-02	3.799e-03	2.698e-17	4.627e-02
CPU	6.599	0.499	0.920	0.874	1.014



(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j)

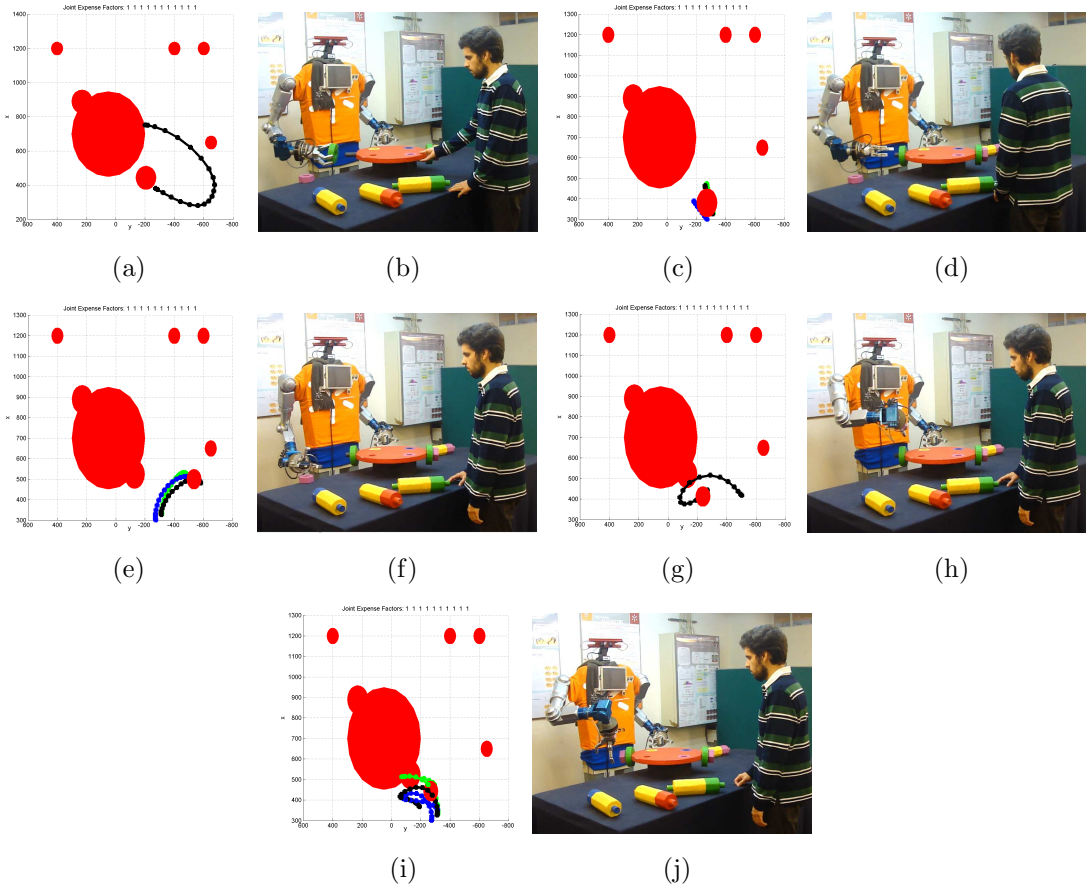
Figure 6.7: Step 1.1 to 3.1 of the Toy Vehicle construction sequence.

The table 6.8 and the figure 6.7 show that the IPOPT is able to find an optimal solution for all the first five bounce posture selection problems of the Toy-Vehicle sequence in a reasonable time for Human-robot interaction. The times to obtain the solutions for all problems range between 0.499 seconds and 1.014 seconds. As visible in the table 6.8, the time took to solve P1.1b is almost 7 seconds (6.599 seconds) and is justified because as it was stated above, grasping an object from the table is a demanding challenge comparatively to other movements. Also note that movement P1.1b is the only calculating 10 joints once it is related to a grasping movement with a side grip thumb up configuration (in which one must calculate the spread of the hand), increasing even more the size of the problem. The problems seen in table 6.8 are of medium dimension from P1.2b to P3.1b, considering the number of constraints (M_{eq}, M_{ineq}) and variables (N) which is between 100 and 1000, the P1.1b is a large scale problem. One may say that the movement P1.2b is the easiest to solve the non-linear optimization problem, as it is seen by the value of the CPU. This is understandable once it is a simple movement of asking for an object, without obstacles nearby. The plots obtained in figures 6.7a, 6.7c, 6.7e, 6.7g and 6.7i show that the trajectories generated by the movement planning are smooth and target directed in line with human movements.

The next table presents the numerical results for bounce postures related to movements from P3.2b to P4.3b, in which the related data is presented and afterwards explained.

Table 6.9: Numerical results for **P3.2b-P4.3b** subproblems

	P3.2b	**P3.3b**	**P4.1b**	**P4.2b**	**P4.3b**
Arm	Right	Right	Right	Right	Right
N	7	9	10	7	10
N_T	20	10	20	20	20
M_{eq}	0	0	0	0	3
M_{ineq}	639	546	1521	639	1281
n_{eq}	0	0	0	0	0
n_{ineq}	2290	2995	8812	2290	7228
Obj	1.616e+000	3.493e-017	1.284e-002	2.11e-2	1.093e-1
CPU	2.012	1.232	5.039	1.029	3.073



(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j)

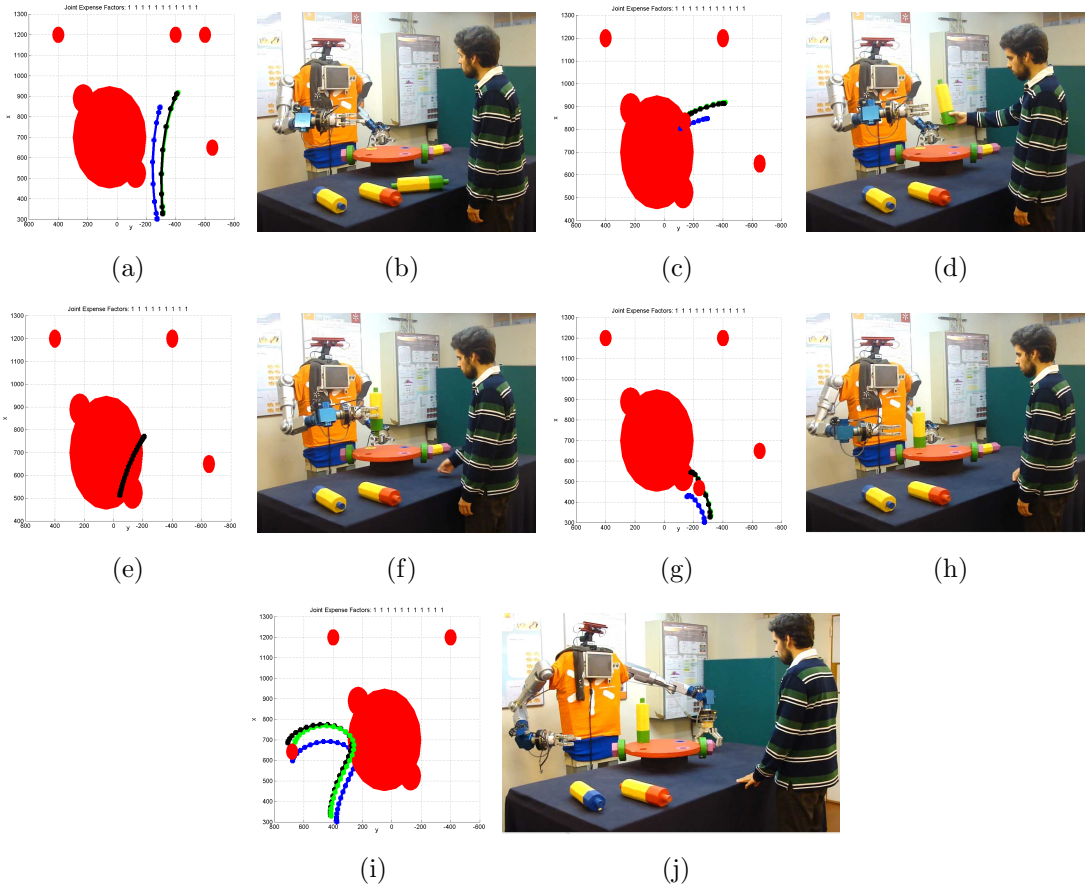
Figure 6.8: Step 3.2 to 4.3 of the Toy Vehicle construction sequence.

The table (6.9 together with the figure 6.8 show that the IPOPT is capable of finding an optimal solution for all these five bounce posture selection problems. The times to obtain the solutions for all problems range between 1.029 seconds and 5.039 seconds. Once again the movement corresponding to grasp an object from the table 6.8f is the most demanding also note that is calculating the movement for 10 joints (side grip thumb up). The problems seen in table 6.9 are of medium dimension for P3.2b, P3.3b and P4.2b, considering the number of constraints (M_{eq}, M_{ineq}) and variables (N) which is between 100 and 1000, the P4.1b and P4.3b problems are of large scale. One may verify that the movement P4.2b is the easiest to solve the non-linear optimization problem, as it is seen by the value of CPU times. Once again the plots obtained in figures 6.8a, 6.8c, 6.8e, 6.8g and 6.8i show that the trajectories generated by the movement planning are smooth and target directed in line with human movements.

The next table presents the numerical results for bounce postures related to movements from P5b to P7.1b, in which the related data is presented and afterwards explained.

Table 6.10: Numerical results for **P5b-P7.1b** subproblems

	P5b	**P6.1b**	**P6.2b**	**P6.3b**	**P7.1b**
Arm	Right	Right	Right	Right	Left
N	9	9	7	9	9
N_T	10	10	30	10	30
M_{eq}	0	0	0	0	0
M_{ineq}	651	606	1490	651	2601
n_{eq}	0	0	0	0	0
n_{ineq}	2995	2900	6554	3660	14875
Obj	4.047e-5	5.632e-15	3.088e-13	4.113e-3	2.332e-1
CPU	0.764	0.484	2.387	1.060	8.315



(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j)

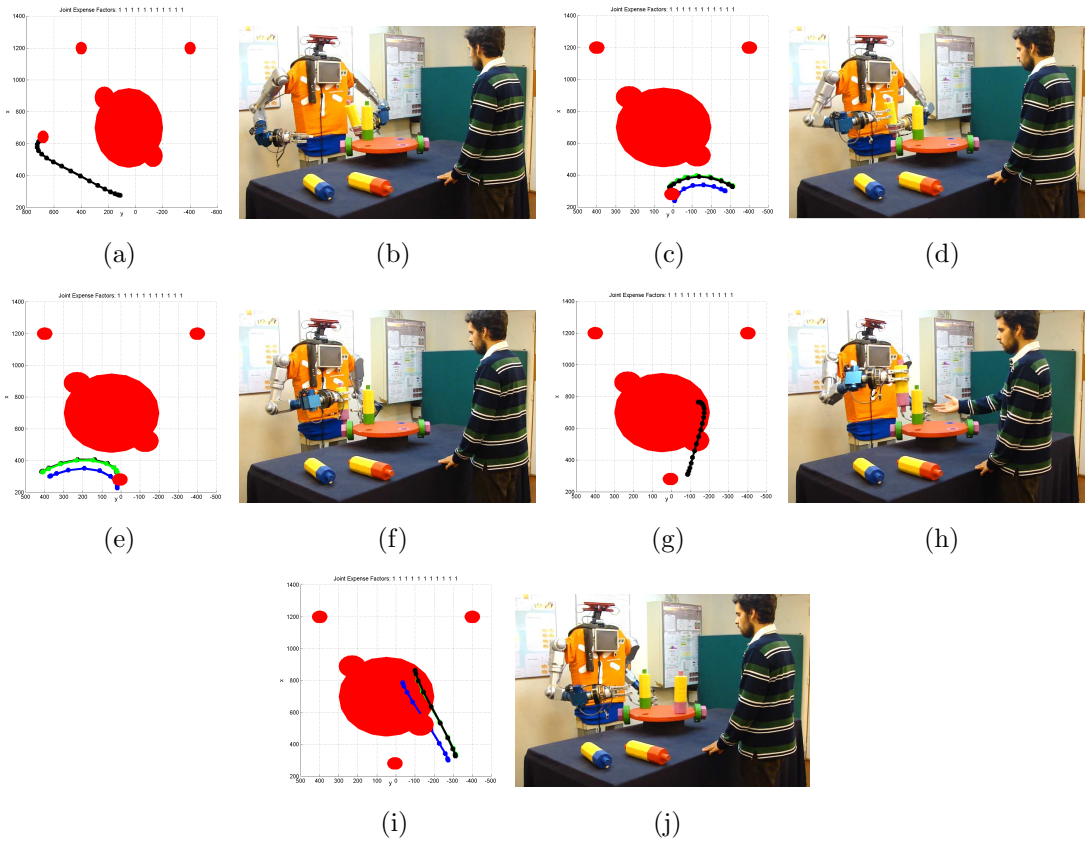
Figure 6.9: Step 5 to 7.1 of the Toy Vehicle construction sequence.

The analysis of table (6.10 and figure 6.9) allow verifying that the IPOPT can find an optimal solution for all these five bounce posture selection problems. The times to obtain the solutions for all problems range between 0.764 seconds and 2.387 seconds. As usual the movement corresponding to grasp an object from the table, which in this case is related to grasp the column 3 with the left arm (see fig. 6.9j) is the most demanding, being computed in 8.315 seconds despite using 9 joints. This time is explained because of the hardness to reach position and orientation of the column 3. The problems seen in table 6.10 are of medium dimension for P5b, P6.1b and P6.3b, considering the number of constraints (M_{eq}, M_{ineq}) and variables (N) which is between 100 and 1000, the P6.2b and P7.1b problems are of large scale. The value of n_{ineq} for P7.1b also helps explaining why it requires such CPU times. One may check that the movement P6.1b is the easiest to solve the non-linear optimization problem, as it is seen by the value of the CPU time. This is reasonable since the "grasp from human hand" movement has a short, direct trajectory as it is seen in figure 6.9c and has no obstacles nearby. The ask movement P5b is not as easy to minimize by the movement planning as the other "Ask" movement because at this point of the construction both nut and wheel are already inserted in the base and therefore are very close to "Ask" trajectory. Once again the plots obtained in figures 6.9a, 6.9c, 6.9e, 6.9g and 6.9i show that the trajectories generated by the movement planning are smooth and target directed in line with human movements.

The next table presents the numerical results for bounce postures related to movements from P7.2b to P7.6b, in which the related data is presented and afterwards explained.

Table 6.11: Numerical results for **P7.2b-P7.6b** subproblems

	P7.2b	**P7.3b**	**P7.4b**	**P7.5b**	**P7.6b**
Arm	Left	Right	Left	Right	Right
N	7	9	9	7	9
N_T	20	10	17	24	40
M_{eq}	0	0	0	0	0
M_{ineq}	862	651	651	662	651
n_{eq}	0	0	0	0	0
n_{ineq}	3696	3660	3660	2668	3660
Obj	4.461e-3	5.583e-3	1.174e-9	1.939e-2	5.571e-3
CPU	0.562	0.749	0.530	0.734	1.186



(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j)

Figure 6.10: Step 7.2 to 7.6 of the Toy Vehicle construction sequence.

By interpreting table (6.11 and figure 6.10)it is reasonable to verify that IPOPT accomplished to find an optimal solution for the last five bounce posture selection problems. The times to obtain the solutions for all problems range between 0.530 seconds and 1.186 seconds. The times obtained for these problems is relatively low and stable, the reason is that there is no movement for grasping from table. The problems seen in table 6.11 are of medium dimension considering the number of constraints (M_{eq}, M_{ineq}) and variables (N) which is between 100 and 1000. Once again the plots obtained in figures 6.10a, 6.10c, 6.10e, 6.10g and 6.10i show that the trajectories generated by the movement planning are smooth and target directed in line with human movements.

6.6 Conclusions for the Toy-Vehicle structure

The results explained in section 6.5 showed that it is possible to integrate AMPL-IPOPT in the movement planning with the purpose to solve the nonlinear optimization problems that arise when modeling the entire human-like trajectory of an anthropomorphic robot arm and hand, including obstacle avoidance. Furthermore this appropriated CPU times obtained for real-time implementation have demonstrated that its implementation is possible since it allows for a fluent asynchronous bimanual manipulation given that the movements are planned and executed at an acceptable time for the human-robot joint construction tasks.

It is worthwhile noting that the values obtained for the "Shorter Sequence" presented at NOA 2013 have different values for the same movements visible in the "full sequence". The main reason for this, is related to some adjustments afterwards in order to obtain a more robust planning for the full sequence. Also the disposition of the obstacles change the model and further values of the variables obtained in the output.

Chapter 7

AROS Movements - *Space-Station* Information, Sequence of Construction and Results

7.1 Space-station Workspace

Similarly to the *Toy Vehicle* scenario, (see section 6.2), the *Space-Station* scenario also presents a well defined workspace in which both human user and **ARoS** are allowed to perform movements. Figure 7.1 displays the *Space-Station* scenario fully mounted with an external plane in $\widehat{\mathbf{z}}$ separating the human's from the robot's workspace.

Once again similarly to what happens in the *Toy-Vehicle* scenario, in order to have an active sharing of tasks between both actors, there are some situations in which they may enter inside each other workspaces. This situation happens when: **1)** the human performs support tasks inside the robot workspace. He/she is not starting the main task but his/her movements are directly supporting the robot (e.g. grasping and securing other objects that are needed in order to make

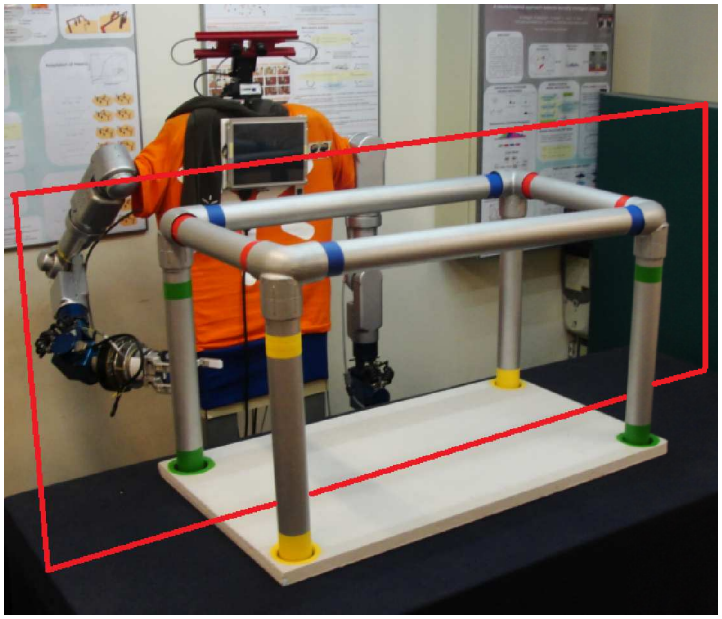


Figure 7.1: Workspaces of both human user and **ARoS** defined by the red plane in the *Space Station* scenario.

it possible to finish the task); **2)** the robot is asking for one object, in the moment of grasping the object might be out of its workspace.

7.2 Grip Types

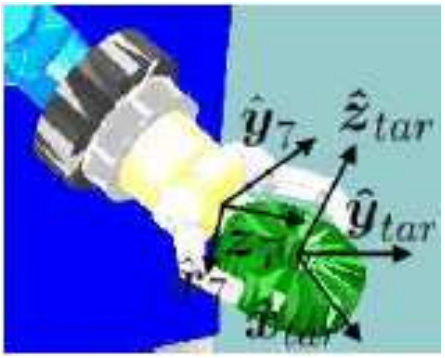
In this section it will be described which grip types were used in order to accomplish the proposed task in the *Space Station* scenario (described in section 7.3.1). Once again, a *precision grip* with the properties established by Napier (1956) will be used as well as Napier's approach to characterize different grip types. The tasks performed in the *Space Station* scenario use two different grip types as seen on table 7.1: **1)** Side Grip thumb right; **2)** Side Grip thumb left.

The figure 7.2 shows two possible positions and orientations of the gripper in order to make it possible to grasp the objects in the *Space-Station* construction scenario. Note that in figure 7.2a we have 2 fingers turned to the human and the thumb towards **ARoS** and in figure 7.2b we have the thumb turned to the human and two fingers towards **ARoS** accordingly to the side grip used.

Table 7.1: Relation between Grip Type, Rotation of the hand, Angles and the Orientation of the Object. As may be seen in the 2^{nd} column there are two different orientations of the hand that can be called upon in order to grasp objects on the table (last column).

Grip type (precision grasp)	Rotation Hand	Angle (deg)	Orientation of the object	Grip code
Side grip - Thumb left	$\widehat{\mathbf{x}}_{tar}$	0	$\widehat{\mathbf{z}}_{tar}$	111
Side grip - Thumb right	$\widehat{\mathbf{x}}_{tar}$	180	$\widehat{\mathbf{z}}_{tar}$	112

(a) Side Grip Thumb right - Used in the *Space Station* scenario.



(b) Side Grip Thumb left - Used in the *Space Station* scenario.

Figure 7.2: Grip types (Thumb left and right) used in the tasks performed in the *Space Station* scenario.

It is important to keep in mind that for the right arm, gasping objects with *side grip thumb left* (2 fingers towards the human, the thumb toward the robot) is easier to perform because it corresponds to the natural orientation of the right hand.

However, the left arm doesn't have its hand symmetrical to the right hand. This means that when the left arm grasps objects with *side grip thumb left* (see fig. 7.2b), as a matter fact the 2 fingers are towards the robot and the thumb is towards the human. In order to obtain a more natural movement the object must be grasped with a *side grip thumb right* configuration (see fig. 7.2a).

7.3 *Space Station* brief explanation and sequence of construction

In this section the sequence of construction to accomplish the task is going to be analyzed. The structure resembles a space station structure (see fig. 7.3) that consists of a rectangular base with four holes, having one hole in each of the four vertices of the base. A column is placed over each hole of the base (there are two green columns for two green holes and two yellow columns for two yellow holes). At this point there is a base with four columns vertically placed in each vertice. Afterwards, each column will be linked with horizontally placed columns linking each vertically placed column to another. Once the base is rectangular we have two smaller columns (red columns) and two longer columns (blue columns).

It is not the objective of this work to have a movement planning for mounting the *Space Station* structure as seen on figure 7.3. In fact the intention is to show how an active cooperation of movements can enhance the capability to perform a complex task. Accordingly, in figure 7.5 there is a description of the objects used in the chosen task: a rectangular base, one green column, one yellow column and one red column.

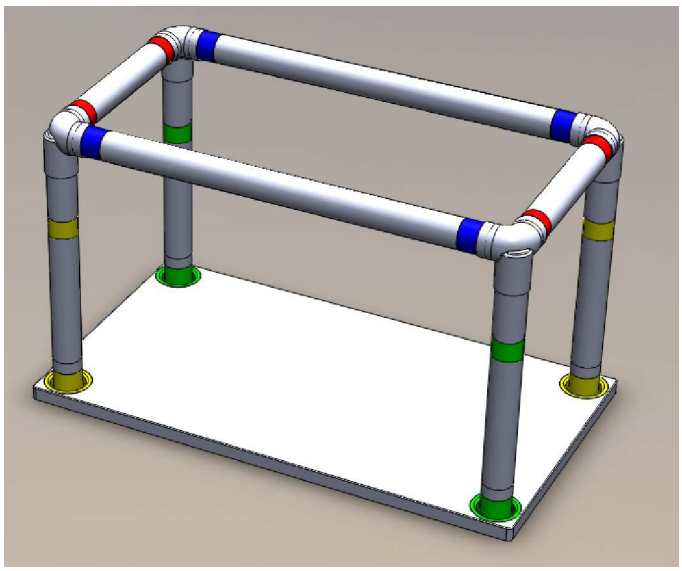


Figure 7.3: *Space Station* fully mounted.

7.3.1 *Space Station* sequence of movements used for the construction

After the presented description of the working scenario, one is able to proceed into the explanation of the sequence of movements implemented. Note that this scenario starts with a Column Green already placed in its final position.

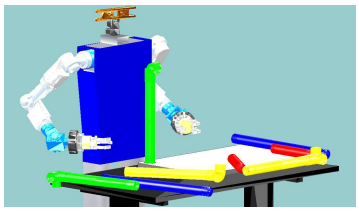
Step 1 - Reach Column Green Side Grip with Right arm (RCg_SG_R) - In this step (see fig 7.4b) **ARoS** must move its right manipulator towards the green column, which is vertically placed over the rectangular base. Note that the grip configuration used is s*ide grip thumb left*. This move may be considered a support movement since its purpose is to keep the column in its place without oscillations while the next movements are performed;

Step 2 - Reach Column Red from Human hand Side Grip with the left arm - In this step, **ARoS** must move its left arm towards the human workspace (it is a slight movement towards the human), over the black table, in order to reach and grasp the red column that is being handed over by the human user (see fig. 7.4c). Note that the grasping is also done with a *side grip thumb right* configuration by the left arm. It is also important to notice that the red column is being grasped by its upper tip (see fig. 7.4d) so that the human can grasp the lower tip and help **ARoS** perform the next movement. Having the hand near the point of insertion of the red column into the green column will also improve the accuracy of the insertion;

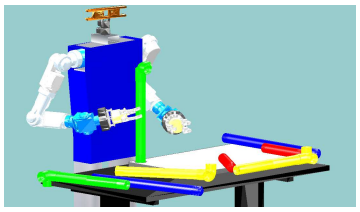
Step 3 - Insert Column Red into the Column Green, Side Grip thumb right with the left arm - In this step (see fig. 7.4e), **ARoS** has both hands grasping the columns: the right arm is holding the green column and the left hand is grasping the red column. The movement now is from the Column Red towards the Column Green, moving the Red Column into the upper tip of the Green Column with the left arm, while holding the Green Column with the right arm. During the movement of insertion, while **ARoS**

left arm is grasping and moving the red column by its upper tip, the human user performs the same task by grasping the column by its lower tip and entering the robot's workspace;

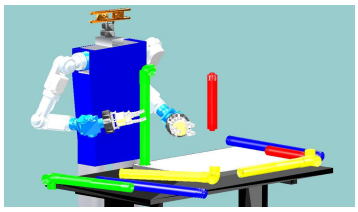
Step 4 - Human Inserting Yellow Column - In this final step (see fig. 7.4f) the human inserts the yellow column in its final position (inside his/her workspace). During this movement the human must be aware of the Red Column once this column must be placed at the upper tip entrance of the Yellow Column (the entrance is parallel to the Red Column). While the human is performing this task, **ARoS** is holding the red column with the left arm and the green column with the right arm. This action is really important to keep the stability of the structure while the human is completing the task.



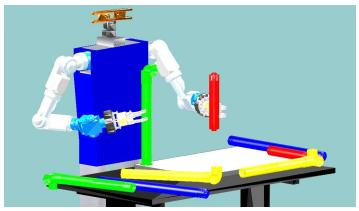
(a) Green Column in final position



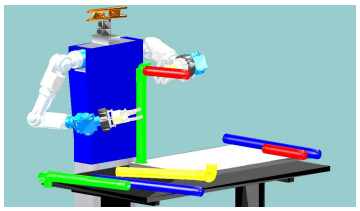
(b) Step 1-Grasp green Column



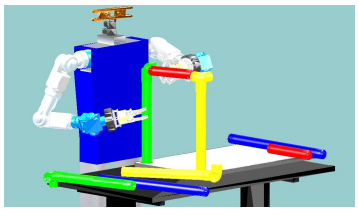
(c) Step 2.1-Red Column



(d) Step 2.2-Grasp Red column



(e) Step 3-Insert Red Column



(f) Step 4 -Human Insert Yellow Column

Figure 7.4: Matlab® Simulation - Sequence of movements obtained during the simulated construction of the *Space-Station*.

To provide a better understanding of the sequence of movements performed in cooperation by the human user and **ARoS**, a sequence of assembly developed

using the SolidWorks® 2012 is represented in figure 7.5:

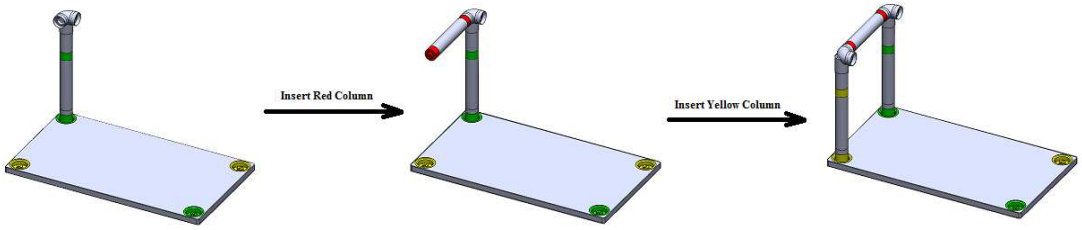


Figure 7.5: *Space Station* assembly sequence used to show advantages of bimanual and simultaneous assembly by both human user and **ARoS**. This scenario was designed in SolidWorks® 2012.

7.4 Results on *Space-Station* bimanual construction scenario

This section will focus on analyzing the sequence of movements implemented in the Space-Station joint construction generated by the movement planning in a human-robot joint construction task. Descriptions of the sequence of movements are presented in table 7.2, while the results will be shown in tables 7.3 and 7.4.

The main focus considering this joint construction is to show how bimanual human robot cooperation allows for more complex manipulation of objects, in a time frame allowing real time interaction.

Besides using the same framework (similar mathematical models and same simulator) as the one used for the Toy Vehicle sequence, new tunings had to be conducted on the simulator and in the AMPL's mathematical models with the purpose to endow the movement planning with the required robustness to complete the new movements.

All optimization problems, **P#a** (final postures) and **P#b** (bounce postures), were coded in AMPL modeling language and solved using IPOPT 3.11. The numerical results were obtained using an Intel(R) Core(TM) i5 M460@2.53GHz

processor running Windows 7 64-bits with a ATI Mobility Radeon HD 5650 video card and 4GB of Ram Memory.

7.4.1 Problems description

More specifically, **ARoS'** movements for this sequence are the following:

P1: Reach to grasp the green column already inserted on the base with the right arm, and hold it;

P2: Reach to grasp a red column from the human with the left arm;

P3: Insert the red column on the top of the green column, while being helped by the human user;

P4: Holding both green and red column while the human inserts the yellow column on the base and connects it to the red column.

Table 7.2 presents details on each of the movements described above.

Table 7.2: Problem Description. Here is visible both **Pa** and **Pb** sub-problems of the "Space-Station" sequence in which Pa is for final posture selection problems (final value of all the calculated joints) and Pb is for bounce posture selection problems (sequence of postures calculated to reach the previously obtained final posture - Pa)

Problem	Movement	Target object	Other objects in the workspace	Pa	Pb	Arm used
1	Reach-to-grasp Green Column	Green Column	Table, base, green column	**P1a**	**P1b**	Right
2	Reach-to-grasp Red Column	Red Column	Table, base, green column	**P2a**	**P2b**	Left
3	Insert Red Column	-	Table, base, green column	$\mathbf{P3a}_1$ $+\mathbf{P3a}_2$	**P3b**	Left
4	Hold Green and Red Columns	-	Table, base, green column	-	-	-

7.4.1.1 Parameters used in this sequence

For the implementation of the sequence related to the Space Station scenario, several parameters have been set. These parameters are the following:

- Movement duration is $T = 1$. Note that this is a unitary value and after having the trajectory planned the effected duration is adapted in order to limit the velocity of the joints;
- Precision of orientation of the end-effector $\delta = 2.0e-3$ (for **P1a** $\delta = 5.0e-2$). The margin of error related to the hand orientation (δ) of P1a is more relaxed than for the other posture problems because as verifiable in figure 7.6b in this situation the arm is only expected to hold the green column. Regarding this fact the orientation of the hand is of high importance;
- Joint expense factors have a default value of $\lambda_k = 1$ for $k = 1, \ldots, n_j$;
- IPOPT was run with the default options: Tol $= 1.0e-6$ and an acceptable tol $= 10.0e-6$. Those are related to error margin accepted for the optimality conditions of the problem in which a solution is considered optimal. Second order derivatives were approximated using a limited-memory method named Broyden-Fletcher-Goldfarb-Shanno (L-BFGS);
- AMPL presolve was set to off in order to maintain the original equations. The presolve setting simplifies the equations of the model, but it was observed that the new equations obtained did not produced the expected behavior.

7.4.1.2 Posture problem results

Tables 7.3 and 7.4 show the numerical results of this sequence. In these tables the following variables are presented:

- **Number of variables,** N directly affects the dimension of a problem and is related to the number of joints being calculated;

- **Number of equality constraints,** M_{eq}, is related to constraints of the problem that are defined as equalities;
- **Number of inequality constraints,** M_{ineq}, is related to constraints of the problem that are defined as inequalities;
- **Number of non-zero elements in equality constraint Jacobian,** n_{eq}, is directly dependent of the equality constraints and affects the dimension of the problem;
- **Number of non-zero elements in inequality constraint Jacobian,** n_{ineq}, is directly dependent of the inequality constraints and affects the dimension of the problem;
- **Objective function value (Obj)**, is the minimum value found for the objective function, while respecting the constraints of the problem, which means, having an acceptable value that verify the optimality conditions of the problem. The smaller optimality value the minor becomes the constraint violation obtained. It is considered a requirement that the value of the constraint violation is below the tolerance defined in the IPOPT's parameters;
- **Computational time in seconds, CPU,** is affected by the dimension of the problem and the difficulty of minimizing the objective function subject to the set constraints on its variables. This may be related to the complexity of the workspace and more or less computationally demanding position of the target to be grasped.

Table 7.3: Numerical results for **P1a**-**P3a**$_2$ subproblems

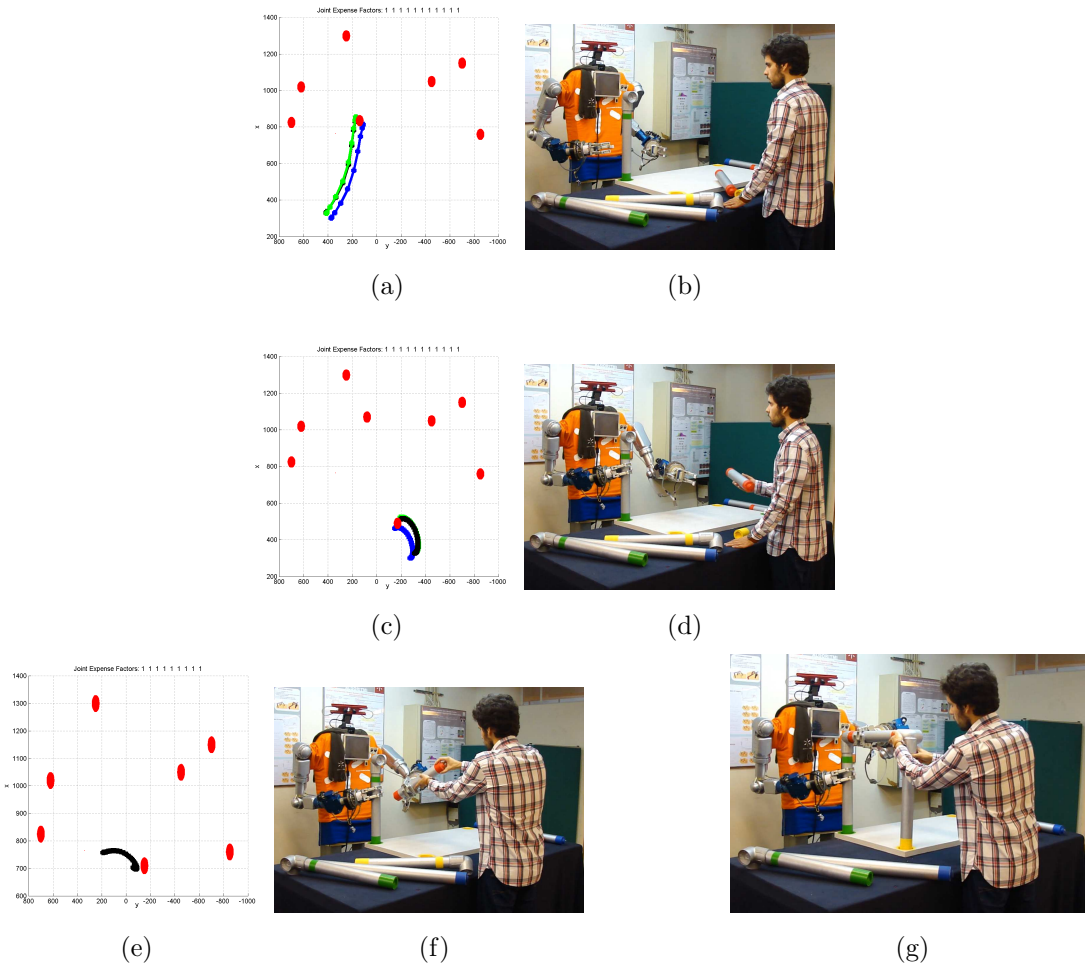
	P1a	**P2a**	**P3a**$_1$	**P3a**$_2$
Arm	Right	Left	Left	Left
N	7	7	7	7
M_{eq}	0	0	0	0
M_{ineq}	15	20	31	31
n_{eq}	0	0	0	0
n_{ineq}	91	112	171	171
δ	0.05	0.002	0.002	0.002
Obj	4.069e-001	1.064e+001	1.562e+000	2.330e+000
CPU	0.187	1.061	0.468	1.014

Table 7.3 presents that the IPOPT is adequately capable of finding an optimal solution for all the four final posture selection problems in a reasonable time for Human-robot interaction. The times to obtain the solutions for all problems range between 0.187 seconds and 1.061 seconds. There is a considerable difference of calculation time between the final posture P2a and P1a. The main reasons that explain this occurrence are related to the fact that arm is more stretched in order to grasp (limiting the number of feasible solutions) the object during the P2a problem, and the fact that the target is grasped with a side grip thumb left configuration instead of the easier side grip thumb right configuration. In order to maintain the human like feature of the movement it was decided to sacrifice the time performance over the human likeness (see more related information in section 7.2). Every problem seen in table 7.3 are of medium dimension from P2a to P3.a$_2$b, considering the number of constraints (M_{eq}, M_{ineq}) and variables (N) which is between 100 and 1000, the P1a is of small scale. These numbers are very small because the movement planning is only considering the green column inside the **ARoS'** workspace. Note that the origin of the other objects lying in the scenario are outside the **ARoS** workspace. The main practical reason to define the problem in this way is to ease the computation and due to the fact that in this situation all the tasks are performed above the rectangular base. If the joint task requires to grasp an object from the table then the premise above described would not be viable and the objects should be considered closer to

ARoS in order to better simulate a real situation.

Table 7.4: Numerical results for **P1b-P3b** subproblems

	P1b	**P2b**	**P3b**
Arm	Right	Left	Left
N	9	9	7
N_T	30	10	30
M_{eq}	0	0	0
M_{ineq}	1401	381	711
n_{eq}	0	0	0
n_{ineq}	7560	1760	2717
Obj	1.569e-002	1.780e-003	1.612e-002
CPU	1.841	0.281	2.060



(a) (b)

(c) (d)

(e) (f) (g)

Figure 7.6: Steps for the construction of the Space-Station. In fig. 7.6g is visible the final stage in which **ARoS** is holding the structure.

Table (7.4 and figure 7.6)show that IPOPT is able to find an optimal solution for all the three bounce posture selection problems of the Space Station sequence in a reasonable time for Human-robot interaction. The times to obtain the solutions for all problems range between 0.281 seconds and 2.060 seconds. As presented in the table 7.4, the problems P1b and P3b are more demanding problems (in relation to P2b). This is justified by the number of time instances done during the movement planning for the trajectory (30 steps), once these two must be more precise than the movement of grasping from the human hand (10 steps) also, the dimension of the problems is different. Every problem seen in table 7.4 are of medium dimension for P2b and P3b, considering the number of constraints (M_{eq}, M_{ineq}) and variables (N) which is between 100 and 1000, the P1b problems is of large scale. The plots obtained in figures 7.6a, 7.6c and 7.6e, show that the trajectories generated by the movement planning are smooth and target directed in line with human movements.

7.5 Implementing movements towards Active Perception of objects

After visualizing and analyzing the Space-Station sequence presented in section 7.3.1, one may notice that the "hole" in which the red column is inserted is facing the human and by consequence cannot be directly seen by the **ARoS'** vision system (see fig. 7.6b). This means that **ARoS** must have a previous knowledge of the object physical characteristics in order to know where the holes are in the green column and to insert the red column.

However, with the purpose of having a more adaptable and human-like solution, the robot must conduct a recognition study of the object, once in unstructured environments there is no previous knowledge of the objects characteristics. Taking this in consideration, **ARoS** should rotate the green column in order to find where the "hole" is located. This process, named *Active Perception*, is an

important feature towards **ARoS** autonomy.

Therefore, an alternate version of the Space-Station sequence was created in order to implement and demonstrate which movements should be implemented to reach the above explained purpose. It is worthwhile note that in order to obtain a functional *Active Perception* of the surrounding objects the vision system must perceive the objects in the robots workspace. In this work there were just implemented the movement premises that allow this particular feature once the vision system is yet to be developed. This new sequence is detailed in table 7.5.

In the previous sequence, in which the rectangular base was parallel to the table, the **ARoS** was not able to rotate the green column sufficiently due to the limitations of the right arm joints (with this configuration the object needs to be rotated about 120 degrees in order to be seen by **ARoS** vision system). In this alternate version, the rectangular base was rotated by 90 degrees, meaning that:

- **ARoS** needs to manipulate, with its right arm, the yellow column (instead of the green column). This consists in grasping, lifting, rotating the object (around 45 degrees in order to be seen by **ARoS** vision system) and placing it back in the base with its initial position and orientation.
- The red column must be grasped from the human with **ARoS'** left arm and then inserted on the side hole of the yellow column (instead of the hole that is facing the human).

Table 7.5: The Space Station sequence with Active Perception.

Problem	Movement	Target object	Other objects in the workspace	Pa	Pb	Arm used
1	Reach-to-grasp Yellow Column and then rotate it	Yellow Column	Table, base, yellow column Table, base, yellow column	$\mathbf{P1a}_1$ $+\mathbf{P1a}_2$ $+\mathbf{P1a}_3$ $+\mathbf{P1a}_4$	**P1b**	Right
2	Reach-to-grasp Red Column	Red Column	Table, base, yellow column	**P2a**	**P2b**	Left
3	Insert the Red Column	-	Table, base, yellow column	$\mathbf{P3a}_1$ $+\mathbf{P3a}_2$	**P3b**	Left

7.5.1 Results obtained from the Active Perception in the *Space Station*

Using the same parameters and settings described in section 7.4.1.1 for the previous Space-Station sequence, one may observe the following results for the final posture selection problems (seen in table 7.6):

Table 7.6: Numerical results for $\mathbf{P1a}_1$-$\mathbf{P1a}_4$ subproblems

	$\mathbf{P1a}_1$	$\mathbf{P1a}_2$	$\mathbf{P1a}_3$	$\mathbf{P1a}_4$	**P2a**	$\mathbf{P3a}_1$	$\mathbf{P3a}_2$
N	7	7	7	7	7	7	7
M_{eq}	0	1	1	1	0	0	0
M_{ineq}	16	15	15	15	5	15	15
n_{eq}	0	6	6	6	0	0	0
n_{ineq}	97	91	91	91	25	91	91
δ	0.05	0.001	0.001	0.001	0.002	0.002	0.002
Obj	1.261e+0	8.025e-1	6.553e-1	5.546e-1	1.059e+1	2.335e+0	2.390e+0
CPU	4.103	0.156	0.109	0.094	0.421	0.156	0.250

As for the bounce posture selection problems, these are showed in table 7.7 and figure 7.7:

Table 7.7: Numerical results for **P1b-P3b** subproblems

	P1b	**P2b**	**P3b**
N	9	9	7
N_T	30	10	30
M_{eq}	0	0	0
M_{ineq}	831	291	477
n_{eq}	0	0	0
n_{ineq}	3696	1216	1573
Obj	8.973e-017	1.402e-002	1.190e-018
CPU	0.827	0.171	1.202

In figure 7.7 it is shown the sequence of movements simulated in Matlab® for the Space-Station sequence towards an Active Perception of the objects. Figures 7.7b, 7.7f and 7.7h display (above perspective) how the movement is going to be executed in **ARoS**.

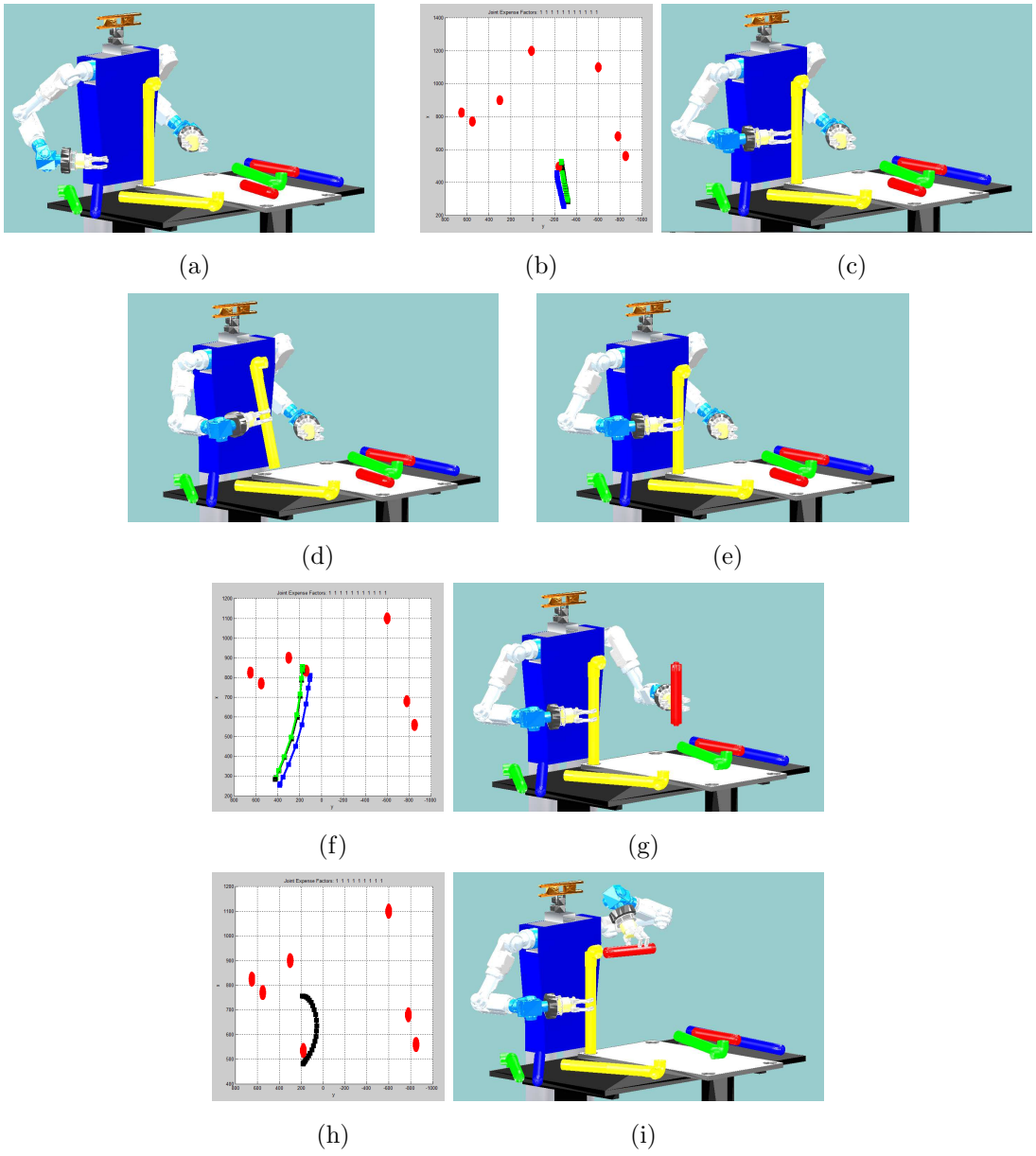


Figure 7.7: Step 1 to 6 of the Space Station, towards Active Perception construction sequence.

The presented results show that IPOPT successfully found an optimal solution for all the final and bounce posture selection problems, in a reasonable time. The CPU times obtained to find the optimal solution range between 0.171 seconds and 1.202 seconds. As may be observed in table 7.6, the time spent to solve $P1a_1$ is

rounding 4 seconds (4.107 seconds), this out of range time may be justified by the illustration 7.7d in which one may see that in order to rotate the yellow column, the manipulator reaches near its joint limits, which is making this particular sub-problem hard to solve.

All the final posture selection problems are of small dimension considering the number of constraints (M_{eq}, M_{ineq}) and variables (N) which is bellow 100. On the other hand, the bounce posture sub-problems are of medium scale, considering the number of constraints (M_{eq}, M_{ineq}) and variables (N) which is above 100

The plots obtained in figures 7.7a, 7.7c and 7.7d show that the trajectories generated by the movement planning are smooth and target directed in line with human movements.

7.6 Conclusions for the Space Station structure

The results explained in section 7.4 showed that it is possible to integrate AMPL-IPOPT in the movement planning with the purpose to solve the nonlinear optimization problems that arise when modeling the entire human-like trajectory of an anthropomorphic robot arm and hand, including obstacle avoidance. Furthermore, appropriated CPU times were obtained for real-time implementation demonstrating that its implementation is possible. More important, in the context of the initial expectations, results showed that the use of asynchronous bimanual manipulation is not only useful, but essential for complex tasks in which both the human and ARoS need to make use of bimanual capabilities in order to mount a structure. This may be seen specially on figure 7.6f, where both actors used two arms, having the robot holding the green and red column and the human inserting the yellow column while adjusting the red column.

Chapter 8

Conclusions and Future Work

8.1 Challenges and compromises

- Adapting the existing Matlab® simulator (ARoS simulator) from unimanual to bimanual movement simulation. This consisted in changing a considerable number of files in the code, creating new global and local variables, adding the new arm (both graphically and mathematically) and adapting the user interface for bimanual control;

- Setting the power supply for the left arm (cables, junction box, emergency stop);

- Understanding the changes in joints between the right and left arm took a long time, due to the complexity of the problem. This consisted in deciding how the left arm should be mounted in relation to the right arm; understanding what joints inverted their direction of rotation; and how it affected the mathematical definition of the left arm (direct and inverse kinematics);

- An obstacle hierarchy system with *(axis-aligned bounding boxes)* AABB's and spheres was previously considered to be integrated with the movement planning, but the optimization process utilized revealed to have sufficient performance without needing it;

- Tuning was needed for the optimization models depending on the **ARoS'** task, because of the type of grip, the shape of the object, the position in the table where the target needed to be grasped, among others. This means that the dimension of the problem slightly varies from case to case;

- During the live tests, the vision system consistently returned wrong values for the z coordinate of the objects. Knowing that the value of the table is fixed, all the objects are given a fixed value of z;

- In a simulation environment, the movement of the arms allowed to correctly grasp the objects in the world. However during the live test, the imprecision of the physical manipulators and the vision system mean that each final posture problem (Pa) needed to be resolved twice - a final and intermediate posture;

- The number of iterations for each Bounce Posture problem is not fixed and varies depending on the type of movement. Movements that required less precision such as reach hand as a request for a piece, handover and grasping from human hand have less iterations than more complex movements (grasp from table and insert);

- During the live tests, the spread of the hand was manually implemented and opens instantly instead of gradually changing. This was necessary due to hardware limitations of the robotic Barrett Hand;

- In some cases during the real time tests, the stereo vision system was not able to accurately find the place of the target, sending wrong locations to the AMPL+IPOPT. Accordingly the output sent by the AMPL+IPOPT had a trajectory in which the final position was slightly off target. This resulted in the collision of the fingers against the object. To avoid this situation some small adjustments were conducted in the output returned by the vision system;

- Another important aspect is related to the fact that an optimal mathematical solution does not mean that posture of the arm is the most adequate for the task. The optimal solution corresponds to the minimal effort for all the joints of the arm and sometimes this may be translated in a posture that is not very human-like. Therefore, in such cases the range of minimal and maximal values for the joints needed to be manually adjusted;

- In this work some Matlab® limitations had to be overcome. The main limitation is related to the fact that Matlab® is not naturally able to work as server and send commands to the hardware. To contour this problem a c++ interface was developed to manage the transmition and reception of information between the Matlab® and **ARoS** hardware.

8.1.1 IPOPT - Matlab® Interface - Attempts and problems

Before using AMPL to model both final and bounce posture problems, the Matlab® Interface with IPOPT was first implemented. The mathematical model was directly modeled in Matlab®. Generally speaking it would works as following:

- **Input**: Requires the objective function, the array with all the constraints and at least the first order derivatives - Jacobian matrices;

- **Minimization**: Given the variables and their upper and lower bounds, the IPOPT tries to minimize the objective function, while respecting the constraints;

- **Output**: Once the optimal solution is found the mathematical variables are directly updated, without requiring an output file like what happens with AMPL and its *.res* files.

The main difference between the above explained Matlab® interface and the AMPL interface is that it does not abstract the problem and requires that every

single constraint and first order derivatives is manually written. This is not specially hard for small scale problems in which the number of constraints and first order derivatives are small. However for the posture based problems this number becomes exponentially high turning the toolbox quickly impractical. Taking in account the practical results from the attempt of using this toolbox in the movement planning problem, one may say that this is not a feasible solution, generating matrices and equations larger than Matlab® display capabilities.

8.2 General Discussion

The aim of this MSc. project was to endow an anthropomorphic robotic platform with unimanual and bimanual movements, firstly by adapting the left arm kinematics and then implementing an optimized movement planning for real-time and *human-like* movements for both arms. This project was implemented for the robotic platform **ARoS**.

There are several possible methods that aboard the problem of unimanual and bimanual object manipulation for anthropomorphic robots. Nevertheless none of the methods are able to fully address all the required specifications needed for fulfilling the desired initial expectations. These were, among others, fluent and human-like arm movements able to simultaneously avoiding obstacles and being target oriented, while maintaining the calculus of the movement planning capable of being implemented in real time situations

As described in the state-of-the-art the methods presented solve some of the required specifications. Some methods have the disadvantage of not finding an optimal solution, such as the *Sampling based motion planning*, ((LaValle, 2006), chapter 5). Other methods suffer from long computational times to solve the movement planning, such as the *Rapidly-exploring Random Trees* (LaValle, 1998) or some optimization methods, like the *Tessellation of joint* (Ozaki and Lin, 1996). Some methods fail to generate human-like movements, such as derivations from the RRT methods, like: *Tangent Space Sampling or First Order Retrac-*

tion (Stilman, 2006, 2010). Lack of human-likeness is also visible in some other methods like the ergonomics criterion (Zacharias et al., 2011), Potential Field Method Khatib (1985), *Attractor Dynamics and Dynamic Potential Fields*. Not considering obstacles is a disadvantage observed in some other methods such as the Extension to basic Attractor Dynamics presented by Reimann et al. (2010) or the *Dynamical Movement Primitives*. Other methods have pre-programmed trajectories which are not viable for real-time applications, as the *Learning by imitation*, *Programming by Demonstration* (Gribovskaya and Billard, 2008) or the *Dynamical Movement Primitives*.

Accordingly, one may verify the complexity of this problem and conclude that those methods are not completely able to solve the problem referred with all its specifications.

In order to aboard this problem in this dissertation the method implemented was based on the Rosenbaum's *Posture-based Motion Planning* described as a non-linear optimization problem, which uses a mathematical abstraction language (AMPL) to model the problem and an optimization solver (IPOPT) to solve it. These methods have been combined with the purpose to take the best features of both with the ultimate intent of address all the initially proposed specifications. After finding which manner would suit best the aims of this project, it was required to adapt it to the tasks of each scenario. The final and bounce posture are chosen accordingly to the "posture-based motion planning" method and the joint configuration of both arms is calculated using the non-linear optimization problem. This step was defined with constraints, tolerances and parameters for every bounce and final postures. One should note why the IPOPT and the ampl were selected among others, like **KNITRO**, **SNOPT** and **LOQO**. The main reason is due to previous tests performed in the unimanual version **ARoS** in which the solvers KNITRO, SNOPT and LOQO were tested and presented higher computational times in most of the tasks (Costa e Silva et al., 2011).

The system was validated for two different sequences, which were both tested

in a MatLab® simulation environment and in live tests. In the Toy Vehicle sequence, the human and the robot had to jointly construct a Toy Vehicle, by handing over, asking and inserting several objects. The results showed that ARoS was able to perform a complex construction sequence, where unimanual and bimanual arm movements could be used to cooperate with the human, generating smooth, fluent and human-like arm postures and trajectories. The CPU times also demonstrated that the system could be used for real-time human-robot interaction.

From this Toy Vehicle scenario, the bimanual system was adapted and validated for a construction sequence of a Space Station structure that features larger components and a more complex assembly. The results obtained from the simulation and live test from this new scenario reinforced the necessity of bimanual arm movements. ARoS was able to assemble a complex part of the structure with both arms, in cooperation with the human partner, which would be much more difficult (or even impossible) without bimanual manipulation. All of this was once again achieved with excellent CPU times, obstacle avoidance and human-like trajectories.

Finally, the movement planning showed good flexibility, because the optimization system was able to calculate both sequences of movements without deep adjustments and tunings. This is because the mathematical models can easily be adapted from one type of movement to another which allowed the system to cover a wide range of applications other than the one tested, manufacturing, children and elder care among others.

8.3 Scientific Relevance

This project has a high relevance for the scientific community once the human-like optimization for bimanual activities isn't accurate. Within the Mobile Anthropomorphic Robotics Laboratory (MARLab) this project is also relevant because the anthropomorphic solution, **ARoS** was only able to execute pre-programmed unimanual tasks .

Part of this work has already received scientific relevance, with results being presented on the paper *"Generating human-like movements on an anthropomorphic robot using an interior point method"*, which was accepted for "AIP Conference Proceedings" and presented at the *International Conference of Numerical Analysis and Applied Mathematics* (ICNAAM 2013) in the symposium related to *Numerical Optimization and Applications* (NOA 2013), in Rhodes-Greece during 21 to 27 of September 2013.

8.4 Future Work

Although the bimanual ARoS platform showed great capabilities and a vast range of application, there is wide range of improvements that can still be made, in order to turn the system more robust.

Firstly, a great improvement can be made in the robustness of the optimization, by not relying on a third-party mathematical software (AMPL), but switching to a Matlab® environment software - OPTI, which would perform the same function as AMPL. The reason for this is that by using AMPL, Matlab® is not able to directly access the value of variables and can only obtain them from an output file generated by AMPL at the end of the optimization process. This becomes quite impractical and the system could become more robust if OPTI was used, because it would run directly into Matlab®. For more information about OPTI, see `http://www.i2c2.aut.ac.nz/Wiki/OPTI/`.

Currently, the system is not able to update the movement planning during

the movement of the arm, which means that changes in the environment (such as the position of the target and/or the obstacles) are not considered. This could be solved by aborting the optimization process if the vision system detects some changes accordingly to a defined threshold, resetting the movement planning.

Another main improvement to be performed would be to switch from asynchronous bimanual coordination to synchronous coordination. Right now, the mathematical models calculated by the optimization system was only modeled to cope with single arm movements. This means that only one movement at a time can be performed between left and right arm. By improving the mathematical model in order to model both arms, would allow to calculate the posture and trajectories for both simultaneously. One should keep in mind that this new mathematical model must be defined in order to keep the real time performance of the system.

Bibliography

T. Asfour, P. Azad, F. Gyarfas, and R. Dillmann, "Imitation learning of dual-arm manipulation tasks in humanoid robots," *International Journal of Humanoid Robotics*, vol. 5, no. 02, pp. 183–202, 2008.

E. Bicho, W. Erlhagen, L. Louro, and E. C. e. Silva, "Neuro-cognitive mechanisms of decision making in joint : a human-robot interaction study," 2011.

E. Bicho, W. Erlhagen, L. Louro, and E. C. e Silva, "Neuro-cognitive mechanisms of decision making in joint action: a human-robot interaction study." *Human movement science*, vol. 30, no. 5, pp. 846–868, 2011.

M. D. K. Breteler and R. G. J. Meulenbroek, "Modeling 3D object manipulation: synchronous single-axis joint rotations?" *Experimental Brain Research*, vol. 168, no. 3, pp. 395–409, 2006.

T. Chaminade, J. K. Hodgins, J. Letteri, and K. F. MacDorman, "The uncanny valley of eeriness," in *SIGGRAPH '07: ACM SIGGRAPH 2007 panels*. New York, NY, USA: ACM, 2007, p. 1.

S. Coradeschi, H. Ishiguro, M. Asada, S. C. Shapiro, M. Thielscher, C. Breazeal, M. J. Mataric, and H. Ishida, "Human-Inspired Robots," *Intelligent Systems, IEEE*, vol. 21, no. 4, pp. 74–85, 2006.

E. C. Costa e Silva, M. F. Costa, E. Bicho, and W. Erlhagen, "Human-Like Movement of an Anthropomorphic Robot: Problem Revisited," in *AIP Conference Proceedings*, vol. 1389, 2011, p. 779.

M. Desmurget and C. Prablanc, "Postural control of three-dimensional prehension movements," *Journal of Neurophysiology*, vol. 77, no. 1, pp. 452–464, 1997.

M. Desmurget, H. Gréa, and C. Prablanc, "Final posture of the upper limb depends on the initial position of the hand during prehension movements," *Experimental Brain Research*, vol. 119, no. 4, pp. 511–516, 1998.

M. A. Diftler, J. S. Mehling, M. E. Abdallah, N. A. Radford, L. B. Bridgwater, A. M. Sanders, R. S. Askew, D. M. Linn, J. D. Yamokoski, F. A. Permenter, B. K. Hargrave, R. Piatt, R. T. Savely, and R. O. Ambrose, "Robonaut 2 - The first humanoid robot in space," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 2178–2183.

M. a. Diftler, T. D. Ahlstrom, R. O. Ambrose, N. a. Radford, C. a. Joyce, N. De La Pena, a. H. Parsons, and a. L. Noblitt, "Robonaut 2 — Initial activities on-board the ISS," *2012 IEEE Aerospace Conference*, pp. 1–12, Mar. 2012.

B. R. Duffy, "Anthropomorphism and the social robot." *Robotics and Autonomous Systems*, vol. 42, no. 3-4, pp. 177–190, 2003.

A. Edsinger and C. C. Kemp, "Two arms are better than one: a behavior based control system for assistive bimanual manipulation," in *Recent Progress in Robotics: Viable Robotic Service to Human*. Springer, 2008, pp. 345–355.

——, "Two arms are better than one: a behavior based control system for assistive bimanual manipulation," in *Recent Progress in Robotics: Viable Robotic Service to Human*. Springer, 2008, pp. 345–355.

C. L. Elsinger and D. A. Rosenbaum, "End posture selection in manual positioning: evidence for feedforward modeling based on a movement choice method," *Experimental Brain Research*, vol. 152, no. 4, pp. 499–509, 2003.

P. Fiorini and Z. Shiller, "Time optimal trajectory planning in dynamic environ-

ments," in *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, vol. 2. IEEE, 1996, pp. 1553–1558.

T. Fong, I. Nourbakhsh, and K. Dautenhahn, "A survey of socially interactive robots: concepts, design, and applications," Carnegie Mellon University Robotics Institute, Tech. Rep. CMU-RI-TR-02-29, 2002.

M. Galicki, "The structure of time-optimal controls for kinematically redundant manipulators with end-effector path constraints," in *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, vol. 1, 1998, pp. 101–106 vol.1.

E. Garcia, M. A. Jimenez, P. G. De Santos, and M. Armada, "The evolution of robotics research," *Robotics Automation Magazine, IEEE*, vol. 14, no. 1, pp. 90–103, 2007.

E. Gribovskaya and A. Billard, "Combining Dynamical Systems control and programming by demonstration for teaching discrete bimanual coordination tasks to a humanoid robot," in *Human-Robot Interaction (HRI), 2008 3rd ACM/IEEE International Conference on*, 2008, pp. 33–40.

Y. Guiard, "Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model," *Journal of motor behavior*, vol. 19, pp. 486–517, 1987.

H. Hoffmann, P. Pastor, D.-H. Park, and S. Schaal, "Biologically-inspired dynamical systems for movement generation: automatic real-time goal adaptation and obstacle avoidance," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on.* IEEE, 2009, pp. 2587–2592.

A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Movement imitation with nonlinear dynamical systems in humanoid robots," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 2. IEEE, 2002, pp. 1398–1403.

T. Inoue, "Future tasks of research in robotics," in *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, vol. 1, 1995, p. 24 vol.1.

I. Iossifidis and G. Schoner, "Dynamical systems approach for the autonomous avoidance of obstacles and joint-limits for an redundant robot arm," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*. IEEE, 2006, pp. 580–585.

F. A. M. Ivaldi, P. Morasso, and R. Zaccaria, "Kinematic networks: a distributed model for representing and regularizing motor redundancy," *Biological Cybernetics*, vol. 60, no. 1, pp. 1–16, 1988.

L. Janssen, M. Beuting, R. Meulenbroek, and B. Steenbergen, "Combined effects of planning and execution constraints on bimanual task performance," *Experimental brain research*, vol. 192, no. 1, pp. 61–73, 2008.

O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, vol. 2, 1985, pp. 500–505.

K. Kim, J. Park, H. Lee, and K. Song, "Teleoperated Cleaning Robots for Use in a Highly Radioactive Environment of the DFDF," in *SICE-ICASE, 2006. International Joint Conference*, 2006, pp. 3094–3099.

H. Kobayashi, Y. Ichikawa, T. Tsuji, and K. Kikuchi, "Development on face robot for real facial expressions," in *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, vol. 4, 2001, pp. 2215 –2220 vol.4.

K. Kondo, "Motion planning with six degrees of freedom by multistrategic bidirectional heuristic free-space enumeration," *Robotics and Automation, IEEE Transactions on*, vol. 7, no. 3, pp. 267–277, 1991.

J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue, "Motion planning for humanoid robots," in *Robotics Research*. Springer, 2005, pp. 365–374.

J. J. Kuffner Jr and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 2. IEEE, 2000, pp. 995–1001.

J.-c. Latombe, "Motion planning: A journey of robots, molecules, digital actors, and other artifacts," *International Journal of Robotics Research*, vol. 18, pp. 1119–1128, 1999.

S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

——, "Rapidly-Exploring Random Trees: A New Tool for Path Planning," Tech. Rep., 1998.

J. Lommertzen, E. C. e Silva, R. H. Cuijpers, and R. G. J. Meulenbroek, "Collision-avoidance characteristics of grasping," in *Anticipatory Behavior in Adaptive Learning Systems*. Springer, 2009, pp. 188–208.

H. G. Malabet, R. A. Robles, and K. B. Reed, "Symmetric motions for bimanual rehabilitation," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, 2010, pp. 5133–5138.

V. Mohan and P. Morasso, "Towards reasoning and coordinating action in the mental space," *International Journal of Neural Systems*, vol. 17, no. 04, pp. 329–341, 2007.

M. N. Mohd Zubir and B. Shirinzadeh, "Development of a high precision flexure-based microgripper," *Precision Engineering*, vol. 33, no. 4, pp. 362–370, 2009.

J. R. Napier, "The prehensile movements of the human hand," *Journal of bone and Joint surgery*, vol. 38, no. 4, pp. 902–913, 1956.

H. Ozaki and C.-j. Lin, "Optimal B-spline joint trajectory generation for collision-free movements of a manipulator under dynamic constraints," in *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, vol. 4. IEEE, 1996, pp. 3592–3597.

N. Padoy and G. D. Hager, "Human-Machine Collaborative surgery using learned models," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 2011, pp. 5285–5292.

D.-H. Park, H. Hoffmann, P. Pastor, and S. Schaal, "Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields," in *Humanoid Robots, 2008. Humanoids 2008. 8th IEEE-RAS International Conference on.* IEEE, 2008, pp. 91–98.

J.-k. Park, "Optimal Motion Planning for Manipulator Arms Using Nonlinear Programming," *Industrial Robotics, Programming, Simulation and Applications*, pp. 256–272, 2007.

P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, "Learning and generalization of motor skills by learning from demonstration," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on.* IEEE, 2009, pp. 763–768.

H. Reimann, I. Iossifidis, and G. Schoner, "Generating collision free reaching movements for redundant manipulators using dynamical systems," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on.* IEEE, 2010, pp. 5372–5379.

D. a. Rosenbaum, R. G. Meulenbroek, J. Vaughan, and C. Jansen, "Coordination of reaching and grasping by capitalizing on obstacle avoidance and other constraints." *Experimental brain research. Experimentelle Hirnforschung. Expérimentation cérébrale*, vol. 128, no. 1-2, pp. 92–100, Sep. 1999.

D. A. Rosenbaum, R. J. Meulenbroek, J. Vaughan, and C. Jansen, "Posture-

based motion planning: applications to grasping." *Psychological review*, vol. 108, no. 4, p. 709, 2001.

L. Ross, "Global investment activities in robotics, nanotechnology and energy," in *Robotics and Biomimetics (ROBIO). 2005 IEEE International Conference on*, 2005, p. xxxviii.

R. Saltaren, R. Aracil, C. Alvarez, E. Yime, and J. M. Sabater, "Field and service applications - Exploring deep sea by teleoperated robot - An Underwater Parallel Robot with High Navigation Capabilities," *Robotics Automation Magazine, IEEE*, vol. 14, no. 3, pp. 65–75, 2007.

S. Schaal, "The new robotics - towards human-centered machines," *HFSP Journal Frontiers of Interdisciplinary Research in the Life Sciences*, no. 2, pp. 115–126, 2007.

Z. Shiller and S. Dubowsky, "On computing the global time-optimal motions of robotic manipulators in the presence of obstacles," *Robotics and Automation, IEEE Transactions on*, vol. 7, no. 6, pp. 785–797, 1991.

E. C. e. Silva, "Reaching, Grasping and Manipulation in Anthropomorphic Robot Systems," 2011.

M. Stilman, "Task Constrained Motion Planning in Robot Joint Space," Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-06-43, 2006.

——, "Global manipulation planning in robot joint space with task constraints," *Robotics, IEEE Transactions on*, vol. 26, no. 3, pp. 576–584, 2010.

V. Tagliasco, "Factors influencing the evolution of robotics," in *Advanced Robotics, 1991. 'Robots in Unstructured Environments', 91 ICAR., Fifth International Conference on*, 1991, pp. 6 –11 vol.1.

K. Tanie, "Human friendly robotics - A challenge to new robot applications," in *1999 IEEE/RSJ International Conference on Intelligent Robots and Systems, 1999. IROS '99. Proceedings*, vol. 2, 1999, p. 609.

R. Tellez, F. Ferro, S. Garcia, E. Gomez, E. Jorge, D. Mora, D. Pinyol, J. Oliver, O. Torres, J. Velazquez, and D. Faconti, "Reem-B: An autonomous lightweight human-size humanoid robot," in *Humanoid Robots, 2008. Humanoids 2008. 8th IEEE-RAS International Conference on*, 2008, pp. 462–468.

Y. Toyoda and Y. Fumihiko, "Optimizing Movement of A Multi-Joint Robot Arm with Existence of Obstacles Using Multi-Purpose Genetic Algorithm," *IEMS*, vol. 3, pp. 78–84, 2004.

T. Tsuji, P. G. Morasso, K. Shigehashi, and M. Kaneko, "Motion Planning for Manipulators Using Artificial Potential Field Approach that can Adjust Convergence Time of Generated Arm Trajectory," *Journal of the Robotics Society of Japan*, vol. 13, no. 2, pp. 285–290, 1995.

A. Ude, A. Gams, T. Asfour, and J. Morimoto, "Task-specific generalization of discrete and periodic dynamic movement primitives," *Robotics, IEEE Transactions on*, vol. 26, no. 5, pp. 800–815, 2010.

N. Vahrenkamp, M. Do, T. Asfour, and R. Dillmann, "Integrated Grasp and motion planning," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, 2010, pp. 2883–2888.

N. Vahrenkamp, M. Przybylski, T. Asfour, and R. Dillmann, "Bimanual grasp planning," in *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on*, 2011, pp. 493–499.

N. Vahrenkamp, D. Berenson, T. Asfour, J. Kuffner, and R. Dillmann, "Humanoid motion planning for dual-arm manipulation and re-grasping tasks," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on.* IEEE, 2009, pp. 2464–2470.

J. Vaughan, D. A. Rosenbaum, and R. G. J. Meulenbroek, "Modeling reaching and manipulating in 2-and 3-D workspaces: The posture-based model," in *Proceedings of the Fifth International Conference on Learning and Development, Bloomington, IN*, 2006.

Y. Wada and M. Kawato, "A via-point time optimization algorithm for complex sequential trajectory formation," *Neural networks*, vol. 17, no. 3, pp. 353–364, 2004.

M. Weigelt, W. Kunde, and W. Prinz, "End-state comfort in bimanual object manipulation." *Exp Psychol*, vol. 53, pp. 143–148, 2006.

F. Zacharias, C. Schlette, F. Schmidt, C. Borst, J. Rossmann, and G. Hirzinger, "Making planned paths look more human-like in humanoid robot manipulation planning," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 1192–1198.

L. Zhang, J. Pan, and D. Manocha, "Motion planning of human-like robots using constrained coordination," in *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*, 2009, pp. 188–195.

R. Zollner, T. Asfour, and R. Dillmann, "Programming by demonstration: dual-arm manipulation tasks for humanoid robots," *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 1, no. Iros, pp. 479–484, 2004.